Bangor University

DOCTOR OF PHILOSOPHY

Modelling human short-term memory for serial order

Preece, Timothy Edward

Award date:
1996

Awarding institution:
University of Wales, Bangor

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 12. Oct. 2024

Modelling Human Short-Term Memory for Serial Order

I'W BDEFNYDDIO YN Y
LLYFRGELL YN UNIG

TO BE CONSULTED IN THE
LIBRARY ONLY

Timothy Edward Preece

Submitted August 1996

School of Psychology,
University of Wales,
Bangor,
Gwynedd, LL57 2DG.

C.P.G. LLYFRGELL U.C.N.W. LIBRARY

ABSTRACT

Serial order is central to much of human behaviour including short-term memory. There exists a wealth of empirical data and a number of attempts have been made at providing a theoretical account of these data. However, no existing model accounts for more than a limited subset of the existing data, and no existing model allows examination of developmental improvement at the same time as covering a wide range of adult data. In the following thesis, two models of short-term memory are presented. The first, a developmental associative recall network, DARNET, uses gradient descent based learning to learn how to perform single trial learning and recall of novel paired associates. The second, a neurobiologically plausible oscillator-based associative recall model, OSCAR, uses Hebbian association to associate items with different states of a reinstatable dynamic control signal, the context. OSCAR is fitted to a range of empirical data including serial position curves, memory span, phonemic similarity effects and item versus order error distributions. Furthermore, it is suggested that if OSCAR is coupled with DARNET, they could provide a developmental account of short-term memory for serial order.

CONTENTS

1.0 Introduction

1.1 Introduction 1
1.2 Mathematical models 1
1.3 Connectionist models 2
1.4 DARNET 3
1.5 OSCAR 3
1.6 Thesis overview 4

2.0 Empirical data on short-term memory for serial order

2.1 Introduction 5
2.2 Serial position curve 6
2.3 Memory span 8
2.4 Phonemic similarity 11
2.5 Baddeley's phonemic similarity effect 14
2.6 Transposition gradients 17
2.7 Item and order errors 21
2.8 Repeated items and the Ranschburg Effect 23
2.9 Summary 25

3.0 Review theories and models of memory for serial order

3.1 Introduction 26
3.2 Bin models 27
3.3 Chaining models 29
3.3.1 Basic chaining model 29
3.3.2 CHARM 32
3.3.3 TODAM 34
3.3.4 Recurrent networks 42
3.4 Hierarchical models 43
3.4.1 Chunking model 43
3.4.2 Perturbation model 47
3.5 Dynamic context models 50
3.5.1 Competitive Queuing model 50
3.5.2 Network model of the Articulatory Loop 57

3.6 Activation gradient models 62

3.7 Summary 66

4.0 Developmental Associative Recall Network (DARNET)

4.1 Introduction 71

4.2 Phase one learning 73

4.2.1 Storage procedure 73

4.2.2 Recall procedure 75

4.2.3 Learning to learn 76

4.3 Simulations during the phase one learning stage 78

4.3.1 Simulation 1: *Learning to learn* 78

4.3.2 Simulation 2: *Effect of varying the learning rate* 81

4.3.3 Simulation 3: *Effect of varying the momentum* 83

4.3.4 Simulation 4: *Trace dimensionality effects* 87

4.3.5 Summary 91

4.4 Phase two learning 91

4.4.1 Simulation 5: *Trace dimensionality effects* 92

4.4.2 Simulation 6: *Paired-associate intralist intrusions* 95

4.4.3 Simulation 7: *A developmental account for cue intrusions* 98

4.5 Summary 101

5.0 Temporal representation and the OSCAR model

5.1 Introduction 103

5.2 Evidence for and properties required of biological clocks 104

5.3 Models of the internal temporal signal 107

5.3.1 Church and Broadbent's storage vector 107

5.3.2 Houghton's control signal 109

5.3.3 Burgess and Hitch's "context" 111

5.4 An oscillator based control signal: the context vector 113

5.4.1 Introduction 113

5.4.2 The context vector 113

5.4.3 Exploring the properties of the context vector 117

5.4.3.1 Simulation 8: *Constant delta theta size* 117

5.4.3.2 Simulation 9: *Non-linear delta theta size* 119

5.4.4 Summary 121

5.5 An oscillator based associative recall memory (OSCAR) 122

5.5.1 Introduction 122
5.5.2 Modelling sequential events with the context vector 123
5.5.3 Learning by Hebbian association 127
5.5.4 Recall and recognition 129
5.6 Simulation 10: *Serial ordered learning and recall* 131
5.7 Additional parameters 135
5.8 Summary 136
6.0 Investigating OSCAR parameter space
6.1 Introduction 137
6.2 Simulation 11: *Increasing inter context spacing* 137
6.3 Effect of decay rate and learning rate 140
6.3.1 Simulation 12: *Introducing a decay parameter* 141
6.3.2 Simulation 13: *Introducing a learning rate parameter* 143
6.3.3 Simulation 14: *Combining decay and learning rate* 145
6.4 Effect of output inhibition 147
6.4.1 Introduction 147
6.4.2 Simulation 15: *Simple inhibitory process during recall* 148
6.4.3 Simulation 16: *Preventing item replication during recall* 152
6.4.4 Simulation 17: *Inhibition and a distractor vocabulary* 156
6.4.5 Summary 160
6.5 Adding noise during learning or recall 160
6.5.1 Introduction 160
6.5.2 Simulation 18: *Adding noise during learning* 161
6.5.3 Simulation 19: *Adding noise during recall* 163
6.5.4 Simulation 20: *Adding noise during learning and recall* 165
6.5.5 Summary 167
6.6 Simulation 21: *Effect of item similarity* 168
6.7 Simulation 22: *Omission errors* 171
6.8 Summary 173
7.0 OSCAR simulations of empirical data
7.1 Introduction 175
7.2 Simulation 23: *Serial position curve* 176
7.3 Simulation 24: *Memory span* 179
7.4 Simulation 25: *Acoustic similarity and memory span* 181

7.5 Simulation 26: *Vocabulary size effects* 183
7.6 Simulation 27: *Transposition gradients* 185
7.7 Simulation 28: *Phonemic similarity effect for alternating lists* 190
7.8 Simulation 29: *Item and order errors* 197
7.9 Simulation 30: *Partial reinstatement of context* 202
7.10 Conclusion 205

8.0 Discussion

8.1 Introduction 208
8.2 Model architectures 208
8.3 DARNET 210
8.4 OSCAR 211
8.5 Accounting for the empirical data 216
8.6 Developmental model of immediate memory for serial order 220
8.7 Conclusion 222

Appendices

Appendix A: Normalising similar items 224
Appendix B: Methods to ensure reliable data generation 226
Appendix C: Further variations on methods for context generation 227
Simulation 31: *Random delta theta size* 227
Simulation 32: *Alternate method for context generation* 228
Appendix D: Convolution and Correlation as associative mechanisms 231

References 233

List of illustrations

Chapter 2

2.1	Serial position curves after one trial for different list lengths	6
2.2	Serial position curves for memory span study	7
2.3	Proportion of lists recalled correctly as a function of list length for different types of stimulus material	9
2.4	Effect of vocabulary size on memory span for letters	10
2.5	Immediate recall of confusable and nonconfusable six-letter sequences	12
2.6	Affect of acoustic similarity on memory span for words	13
2.7	Phonemic similarity effect for alternating lists	16
2.8	Distance functions based upon the order only experiment	18
2.9	Distance functions for a five item list after one, two or three repetitions	19
2.10	Distance function for nonconfusable-confusable list	20
2.11	Order only and item only data for four item list	22

Chapter 3

3.1	Nonassociative theory of memory for a sequence of digits	27
3.2	Associative theory of memory for a sequence of digits	30
3.3	CHARM's method of item encoding, decoding and recognition	33
3.4	Jordan recurrent network and Elman recurrent network	42
3.5	Decoding the first memory code into the three codes representing the chunks	44
3.6	Decoding operation for recalling a nine letter sequence organised in three chunks	45
3.7	Neighbouring items connected to a control element	48
3.8	Reverbatory loop for decay	49
3.9	Inter-layer connections in the basic CQ model	51
3.10	I-node and E-node activations for a seven item sequence	52
3.11	The activation history of phoneme units during the recall of the word /strIng/	55
3.12	Outline of the network model of the Articulatory Loop	59
3.13	Primacy gradient for a six item list	63

Chapter 4

4.1	DARNET storage architecture	74
4.2	DARNET recall architecture	75
4.3	Error score between the output and target item as DARNET learns-to-learn	80
4.4	Effect of increased learning rate during phase one learning	82

4.5 Effect of momentum term during phase one training 86
4.6 Effect of increasing the trace length from four to fourteen elements on the capacity to store and recall four element input vectors 88
4.7 Effect of increasing the dimensionality of the composite memory trace during a phase two paired-associate learning task for both nonconfusable and confusable stimuli 95

Chapter 5

5.1 Five successive states of the 11-element Church and Broadbent storage vector 108
5.2 Cosine similarity curve for the Church and Broadbent storage vector 108
5.3 I-node and E-node activations for a seven item sequence 110
5.4 Cosine similarity for the Houghton I-E node control signal 110
5.5 Cosine similarity for the Burgess and Hitch context vector 112
5.6 Context vector generation from oscillator array 115
5.7 Composition of the eight element context vector, c 115
5.8 Eight bit context vector evolving through eight successive states 116
5.9 End-on and cut-away view of cosine between neighbouring context vectors 118
5.10 End-on and cut-away view of cosine between neighbouring context vectors 121
5.11 Associating items presented at discrete intervals to the contexts that correspond to those moments in time 123
5.12 Similarity of six "highly distinct" contexts 124
5.13 Similarity of six "less distinct" contexts 125
5.14 Open loop implementation of OSCAR 129
5.15 Serial position curve for basic OSCAR network 133
5.16 Distance functions for basic OSCAR network 134
5.17 Non-linear learning rate decaying with serial position, *i* 135

Chapter 6

6.1 Serial position curves for six item lists as inter-context spacing is increased 138
6.2 Mean proportion correct recalls for increasing inter-context spacing 139
6.3 Serial position curve with decay rate parameter 142
6.4 Serial position curve with learning rate parameter 144
6.5 Serial position curve illustrating the combined effects of a non-linear learning rate and decay rate 146
6.6 The effect of a simple inhibitory process during recall 150
6.7 Distance functions illustrating the effect of a simple inhibitory process during recall 151
6.8 Introducing an inhibitory process preventing repeated item errors 154
6.9 Distance functions illustrating the effect of an inhibitory process preventing repeated item errors during recall 155

6.10 Introducing an inhibitory process preventing repeated item errors with the addition of a vocabulary of distractor items during recall 158
6.11 Distance functions illustrating the effect of an inhibitory process preventing repeated item errors with the addition of a vocabulary of distractor items during recall 159
6.12 Adding increasing levels of noise to the Hebbian memory matrix during learning 162
6.13 Adding increasing levels of noise to the Hebbian memory matrix during recall 164
6.14 Adding noise to the Hebbian memory matrix during both training and recall 165
6.15 Distance functions illustrating the effect of adding noise to the Hebbian matrix during both training and recall 166
6.16 The effect of increasing item similarity on serial order performance 169
6.17 The effect of increasing item similarity on distribution of order errors in third position 170
6.18 Proportion of errors that are omissions for each serial position 172

Chapter 7

7.1 OSCAR modelling the serial position curve of Baddeley (1968) 178
7.2 OSCAR memory span compared with Crannell & Parrish (1957) 180
7.3 OSCAR fit to effect of acoustic similarity on memory span (Baddeley, 1966) 182
7.4 Mean proportion of items recalled correctly as a function of size of distractor lexicon 184
7.5 Distance functions for six item lists of nonconfusable items (Henson et al., 1996) 187
7.6 OSCAR distance functions for six item lists of nonconfusable items 187
7.7 Distance functions for six item lists of confusable items (Henson et al., 1996) 189
7.8 OSCAR distance functions for six item lists of confusable items 189
7.9 Serial position curves for Baddeley's (1968, experiment 5) alternating list conditions 191
7.10 Serial position curves for OSCAR's fit of Baddeley's alternating list conditions 193
7.11 OSCAR transposition matrices for six item list of nonconfusable, confusable items 194
7.12 Order only and item only data for four item list (Healy, 1974, three digit delay) 198
7.13 Distance functions for the order only experiment (Healy, 1974, three digit delay) 199
7.14 OSCAR serial position curves for order only and item only experiments 200
7.15 OSCAR distance functions for the order only simulation 200
7.16 Proportion of recall as a function of serial position (ordinal probe) 203
7.17 Serial position curves for item lists with partial reinstatement of learned-contexts 204

Chapter 8

8.1 Developmental oscillator based associative recall model 222

Appendices

C.1 End-on and cut-away view of cosine between neighbouring context vectors 228

C.2 Composition of the eight element context vector, **c** 229

C.3 End-on and cut-away view of cosine between neighbouring context vectors 229

List of tables

Chapter 2

2.1 Six letter arrangements used during phonemic similarity effect experiment 15

Chapter 3

3.1 Composition of the memory trace vector after presentation of the jth pair 33

3.2 Composition of the memory vector M_j after presentation of the jth item 36

Chapter 4

4.1 Storage weights S_{ijk} after training to high degree of accuracy 90

4.2 Mean percentage recall for CHARM 96

4.3 Mean percentage recall for DARNET replication of Metcalfe Eich Simulation 1 97

4.4 Mean percentage recall for confusable items with well and partially trained DARNET 100

Chapter 5

5.1 Three successive states of the modified Burgess and Hitch context signal 112

5.2 Summary of the learning and recall sequence employed by the OSCAR model 122

Chapter 7

7.1 Summary of empirical data addressed by OSCAR 175

7.2 Summary of parameter values for each simulation of the empirical data 207

ACKNOWLEDGEMENTS

I wish to take this opportunity to thank the following: my family, for their support; Emma, for her companionship; Gordon Brown, for his guidance and supervision; and Dylan, Sandy and Pete for lengthy discussions about neural-network modelling. Finally, I must thank Digital Image Design Ltd. for allowing me the time to complete this research.

This research was funded by a University of Wales, Bangor, Scholarship.

CHAPTER 1

Approaches to modelling human memory

1.1 Introduction

The aim of this thesis is to develop new approaches to the modelling of human memory for serial order that combine the best features of existing models whilst avoid their limitations. In this short introductory chapter we briefly introduce the main issues covered in this thesis.

1.2 Mathematical models

A number of recent models of memory such as CHARM (Composite Holographic Associative Recall Model: Metcalfe Eich, 1982, 1985), and TODAM (Theory of Distributed Associative Memory: Murdock, 1982, 1983; Lewandowsky & Murdock, 1989) rely on the mathematical processes of convolution and correlation for learning and recall respectively. These models also employ a distributed representation of information and store all associations in the same central location. We describe them in detail in chapter 3. These models have been successful in modelling a wide range of empirical data (e.g. Lewandowsky & Murdock, 1989) and offer a number of advantages including the ability to perform accurate single trial learning, a resistance to partial damage and gradual unlearning. However, the manner in which the associative and recall mechanisms are defined leads to a number of undesirable properties. In particular, as the associative mechanism is hard-wired into the architecture, so a developmental account of learning is not possible in such models, nor is it clear that they could be extended to provide one.

1.3 Connectionist models

An alternative to the mathematical models described previously are connectionist models such as Rumelhart and McClelland's (1986) back-propagation model, the network model of the articulatory loop (Burgess & Hitch, 1992, 1996) and the competitive queuing architecture (Houghton, 1990, 1994a).

A traditional advantage of connectionist architectures has been their ability to provide a developmental account of learning (Rumelhart & McClelland, 1986). The use of gradient descent learning procedures such as back-propagation (Rumelhart, Hinton & Williams, 1986) provides a mechanism for incremental learning. Connectionist networks also possess the ability to generalise learned behaviour to previously unseen stimuli. Their capacity for storage is not limited in the manner of mathematical models, as the only limit on the capacity of the network is the increase in time required for learning as the storage capacity is increased.

One of the main disadvantages of developmental connectionist networks is their inability to perform single trial learning. However, this may be rejected as a number of recent connectionist models have demonstrated the ability to do so (e.g. Houghton, 1990, 1994a; Burgess & Hitch, 1992, 1996). However, those connectionist models capable of performing single-trial learning fail to provide a developmental account of learning. Furthermore, McCloskey and Cohen (1989) criticise connectionist architectures for their susceptibility to catastrophic interference. This is the inability of a connectionist network to recall previously learned patterns as new patterns are learned (Ratcliff, 1990; Lewandowsky, 1991).

In summary, connectionist frameworks for learning and recall have proved successful in providing a developmental account of learning (e.g. Rumelhart & McClelland, 1986). They possess the ability to generalise learned behaviour to previously unseen stimuli. They have also been applied successfully to the problem of single trial learning (e.g. Houghton, 1990, 1994a; Burgess & Hitch, 1992, 1996). However, traditional connectionist architectures, such as those that implement back-propagation learning,

are susceptible to catastrophic interference (Ratcliff, 1990; Lewandowsky, 1991) and can not perform single-trial learning.

It is our aim to develop a novel architecture that is capable of performing the single-trial learning of the mathematical models as well as providing a developmental approach to learning similar to that provided by the connectionist architectures.

1.4 DARNET

The developmental associative recall network, DARNET, is a connectionist based architecture that learns to perform single trial learning and recall (Brown, Hyland & Hulme, 1994; Brown, Dalloz & Hulme, 1995; Brown, Preece & Hulme, 1995; Brown, Hulme & Dalloz, 1996). We illustrate how DARNET gradually learns to perform accurate single trial learning and recall to a level of performance at least as good as the mathematical models. We then demonstrate how DARNET can provide a developmental analysis of paired-associate recall data.

1.5 OSCAR

While DARNET provides us with a developmental account of single-trial learning, it is not capable of producing serial ordered behaviour. We therefore also develop a novel architecture for serial ordered learning and recall. OSCAR, an OSCillator-based Associative Recall network, is an independently-motivated model that employs item-to-context association as the mechanism for ordered learning and recall. A system of oscillators generates a reinstatable context to which items are associated by Hebbian learning during list presentation. Recall involves reinstating the learned-contexts in order that the items be retrieved from the memory trace. Finally, we demonstrate how the model can reproduce a range of serial order benchmarks including the serial position curve for immediate recall of visually presented stimuli, the phonemic similarity effect for alternating list conditions (Baddeley, 1968, experiment 5) and item and order error distributions (Healy, 1974).

The proposal is to combine both these novel architectures to provide a developmental model of serial order capable of replicating the range of empirical data illustrating the development of short-term memory in children.

1.6 Thesis overview

This thesis is divided into three main sections. The first contains two literature reviews: one which presents significant empirical findings for human immediate memory for serial order (chapter 2); the second which describes a number of models and theories of association and short-term memory (chapter 3). It is observed that a wide range of empirical data has been accounted for by a range of models using different underlying associative mechanisms, e.g. item-to-item association (Wickelgren, 1965a; Jordan, 1986; Lewandowsky & Murdock, 1989), item-position association (Conrad, 1965), item-to-dynamic-control-signal (Houghton, 1990, 1994a; Burgess & Hitch, 1992, 1996) or activation gradients (Page & Norris, 1995).

The second section introduces a novel developmental model of association, DARNET (chapter 4). It is applied to modelling empirical data for paired-associate learning (Metcalfe Eich, 1982). We demonstrate how a developmental account of association is necessary in order to address experimental findings (Brown, Preece & Hulme, 1995).

The third section (chapters 5, 6 & 7) describes a novel model of serial order, an oscillator-based associative network, OSCAR. The influence of a number of free parameters and inhibitory processes is investigated. The model is shown to fit a range of empirical data including the phonemic similarity effect for alternating lists of Baddeley (1968, experiment 5) which has proved impossible for chaining based models to address (Baddeley, Papagno, & Norris, 1991; Burgess & Hitch, 1992; Henson, Norris, Page & Baddeley, 1996).

Finally, chapter 8 presents a general discussion of the implication of the findings presented in this thesis while the appendices contain further investigations of both the effects of normalisation and the context control signal.

CHAPTER 2

Empirical data on short-term memory for serial order

2.1 Introduction

The problem of serial order concerns how the brain encodes, stores and retrieves temporally ordered information. In the following chapter, a number of experimental findings that provide benchmark empirical data for formal models of short-term memory for serial order are discussed.

The first paradigm is the serial position curve. This illustrates how recall performance varies across the length of the list. For visually presented stimuli, the curve is bowed with a primacy effect for early items and last-item recency due to edge-effects (e.g. Baddeley, 1968). Section 2.3 discusses the 'reverse-S' shaped memory span curve (e.g. Crannell & Parrish, 1957). Conrad (1964) observed that acoustic, or phonemic, confusability between letters can account for the type of error produced not only when auditory presentation is used, but also with visual presentation. The affect of phonemic similarity on serial recall and memory span is discussed in section 2.4. In section 2.5, Baddeley's (1968, experiment 5) phonemic similarity effect for lists containing alternately confusable and nonconfusable items is reviewed. The fourth paradigm to be addressed is the distribution of order errors (section 2.6). In section 2.7 we discuss Healy's (1974) observation that as serial position curves of item and order errors are different, different mechanisms may be responsible for each. Finally, in section 2.8, we consider lists that contain repeated items and the Ranschburg effect.

A model of short-term memory for serial order must be able to replicate the majority of the results described in this chapter.

2.2 Serial position curve

Typically during a serial order task, subjects are presented with a number of items such as letters (e.g. Baddeley, 1968) or digits (e.g. Conrad, 1959). These may be presented visually (e.g. Conrad, 1965) or verbally (e.g. Jahnke, 1963). Recall in the correct serial order may be required immediately after the stimulus has been presented (e.g. Wickelgren, 1966) or after a short delay (e.g. Conrad, 1960b).

In a typical immediate serial recall task (Jahnke, 1963), subjects were presented with a number of letters, selected from the nine most frequent English consonants, presented verbally at a uniform rate. Subjects were instructed to memorise the letters and to recall them in order, omitting letters they were unsure of, by filling in blank spaces on an answer sheet. Depending on the number of items presented during learning, subjects were expected to complete recall within 10 to 14 seconds. No letter occurred more than once in any one list. Subject performance was measured by recording the mean proportion of times each item was recalled correctly in its correct serial position and is presented in the serial position curve (figure 2.1).

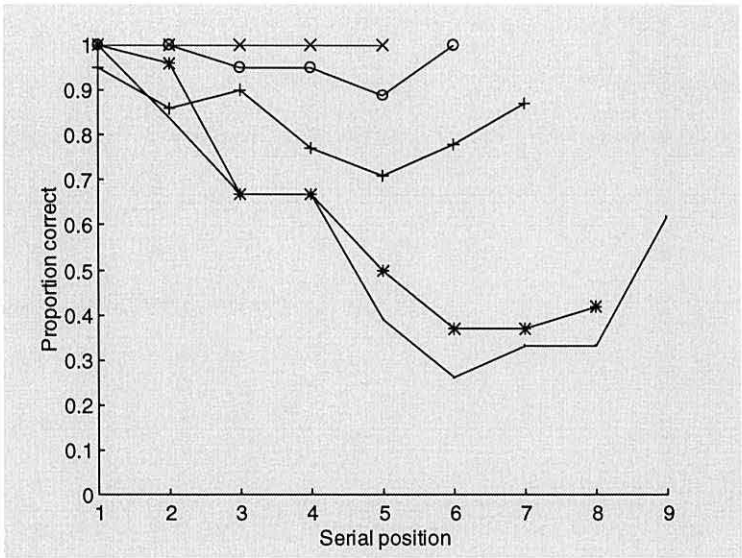

Figure 2.1 *Serial position curves after one trial for different list lengths (Adapted from Jahnke, 1963)*

Although the precise nature of the serial position curve depends upon the experimental details (such as the method of recall and modality of the stimuli), serial position curves

share a common set of features. Overall the curves are 'U'-shaped. There is a *primacy* component: the high degree of performance exhibited by subjects for items extending over the earliest list positions. There is also a *recency* component: an improvement in performance over the last (couple of) serial positions. For visually presented items this will typically involve only the last item and may be attributed to edge effects. The minimum of the serial position curve is usually situated just after the middle of the list. Performance for short lists can be very high (even 100% for each item) however as the list length increases, so the asymmetric bowing of the curve becomes more apparent, the depth of the minimum greater and, overall, performance decreases (Jahnke, 1963).

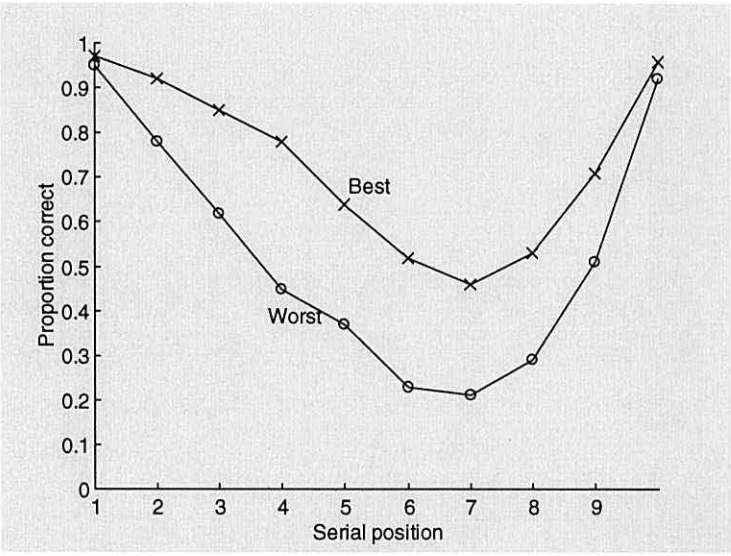

Figure 2.2 *Serial position curves for memory span study (Adapted from Murdock, 1968)*

In one of a series of serial order recall experiments (Murdock, 1968, experiment 6), subjects were presented with a list of 10 digits, presented auditorily at a regular rate, and were required to recall the items in order by filling blank boxes on an answer sheet. Scoring was to the first error and, in contrast to Jahnke (1963), subjects were encouraged to guess when unsure. Every five trials, subjects participated in an alternate memory study.

Murdock's results (figure 2.2) illustrate that subjects also produced the asymmetric bowed serial position curve, with a minimum in the seventh serial position. However, Murdock found that a very large, if brief, recency effect was produced with recall of

the last item being almost 100% in both the *best* subjects and *worst* subjects conditions. The primacy effect extends over the first six or seven items.

Conrad and Hull (1968) confirm that the input modality of stimulus material can affect the form of the immediate recall serial position curve. In an experiment, they presented subjects with seven digit sequences and required that subjects fill their answers, in order, on a blank answer sheet, guessing when necessary. In the visual presentation condition, subjects read the digits silently. In the auditory condition, they read them aloud. Conrad and Hull report that in both conditions, the serial position curve (Conrad and Hull, 1968, figure 1) illustrated that there was an extended primacy effect stretching over the first four items. The minimum in both conditions was in the fifth serial position (recall recorded as 50% correct for the auditory condition, 43% correct for the visual condition). There was a sizeable recency effect for the auditory condition, with performance for the last item approximately 89%. However, in contrast, there was only a slight recency effect in the visual condition, with performance only improving 7% above the minimum in the fifth serial position.

In summary, serial position curves illustrate recall performance in terms of the proportion of trials during which each item was recalled correctly in the appropriate target serial position. In order to eliminate modality effects, we focus on visually presented stimuli (e.g. Conrad & Hull, 1968). For visually presented stimuli, the serial position curve possesses only a small amount of last item recency. Replicating the serial position curve is one of the primary objectives for any of the formal models of serial order memory reviewed in chapter 3. However, until recently, the majority of models of serial order have had great difficulty in reproducing the serial position curve for visually presented stimuli.

2.3 Memory Span

A second fundamental serial-order result is the *memory span function*. This is a record of how the proportion of lists recalled correctly varies as the number of items to be learned, the *list length*, increases.

In a similar fashion as before, subjects are presented with lists of items and required to recall them correctly in the appropriate serial positions. However, only lists recalled completely correctly are scored and hence the memory span curves reflect not the performance *within* the list, as does the serial position curve, but performance as a function of the length of each list. Therefore, the memory span function represents the capacity of short term memory.

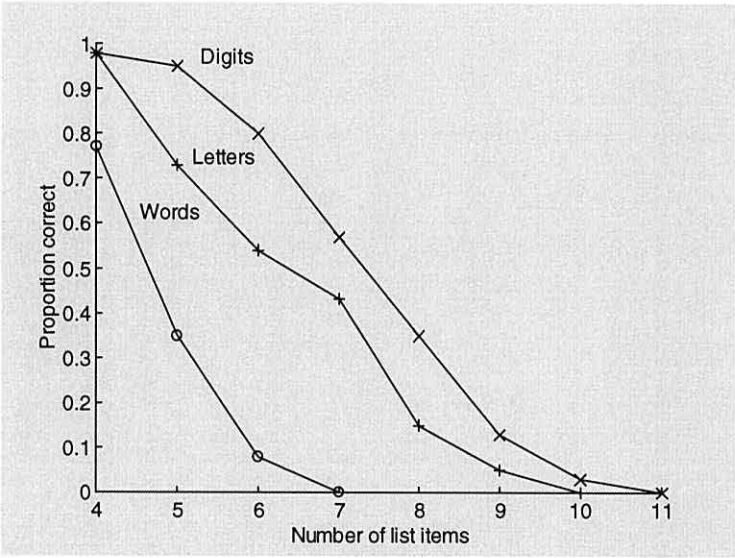

Figure 2.3 *Proportion of lists recalled correctly as a function of list length, for different types of stimulus material (Adapted from Crannell & Parrish, 1957)*

Although the precise shape of the memory span function depends upon experimental details, in particular the nature of the stimuli, the curves do share a number of common attributes. Performance for shorter lists is typically around 100% but decreases, in a 'reverse-S' form towards 0% for longer lists. This is apparent in figure 2.3 where it is clear that the memory span for digits adheres to the 'reverse-S' shape, with performance for four item lists at 100%, dropping to 58% for seven item lists and 0% for 11 item lists. However, it also reveals how the nature of the stimuli affects performance. The memory span function for words is much poorer than that for digits: subjects recall four word lists correctly on approximately 78% of trials, dropping to 0% with lists of seven or more items.

However, there is a related indicator of short-term memory capacity, and that is simply *memory span*. This is drawn directly from the memory span function, and corresponds

to the list length at which subjects recall all items correctly 50% of trials. Therefore, based upon the data presented in figure 2.1, memory span for digits is approximately 7.3 items, for letters approximately 6.2 items and for words, approximately 4.8 items. Clearly, memory span depends upon the nature of the stimuli, although Miller (1956) reports that memory span for arbitrary stimuli varies between five and nine items.

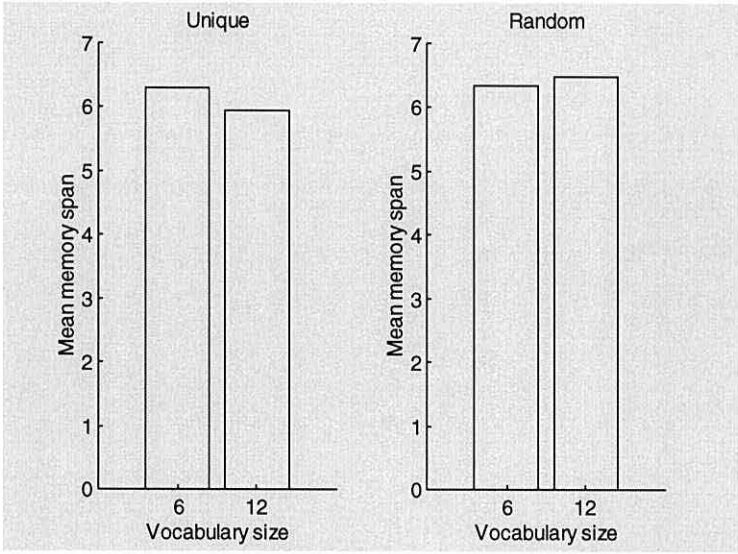

Figure 2.4 *Effect of vocabulary size on memory span for letters (Adapted from Drewnowski, 1980)*

Drewnowski (1980) confirms that memory span depends upon the stimulus material but reports that it does not depend upon the size of the vocabulary of items from which the stimuli are selected. In a memory span experiment, Drewnowski (1980, experiment 1) presented subjects with one of two types of consonant sequence: either *unique*, where no item was repeated, or *random*, where repeated items were permitted. Sequences varied in length and were drawn from either a vocabulary of six or 12 items. Subjects were requested to recall the lists in order and to guess when unsure by selecting an item from the vocabulary of items. Results indicated that there were negligible effects due to vocabulary size (figure 2.4). In an experiment examining the effects of phonemic similarity, Conrad and Hull (1964) found a similar effect for items selected from a vocabulary of three or nine items (figure 2.5).

In summary, the memory span function is a 'reverse-S' shaped curve that illustrates short-term memory capacity. The gradient of the curve depends upon the nature of the

stimuli (Crannell & Parrish, 1957). Memory span (defined as the number of items recalled correctly 50% of trials) is typically approximately seven items (Miller, 1956). The size of the vocabulary from which stimuli items are selected does not affect memory span (Conrad & Hull, 1964; Drewnowski, 1980). Formal models of memory for serial order should ensure that they possess a memory span within the range of the empirical data and are capable of reproducing the memory span function.

2.4 Phonemic Similarity

So far we have not discussed the nature of the stimuli presented to subjects during learning, beyond referring to the differences in memory span observed by Crannell and Parrish (1957). When stimuli are presented visually, confusions occur which are similar to those that would be expected if the stimuli had been presented auditorily (Conrad, 1964). This leads to an increase in *order* errors (Wickelgren, 1965a) and also a reduction in memory span (e.g. Conrad & Hull, 1964; Baddeley, 1966). In the following section we describe evidence that illustrates this, the phonemic similarity effect.

In a serial order recall task, Conrad (1964) presented sequences of six consonants visually to subjects. Subjects, aware that the lists did not contain any repeated items, were told to recall the items in order and to guess at answers if they were unsure by selecting a response from a vocabulary of 10 items, visible throughout the experiment. The vocabulary contained two groups of letters, both with high *within-group* acoustic similarity and low *between-group* acoustic similarity (e.g. *BCPTV* and *FMNSX*). Analysis was limited to single *substitution* errors i.e. only sequences containing one incorrect letter and no other errors. Results were presented in the form of a 10x10 *confusion* matrix, illustrating which letters were written as which responses to which stimuli. In the second part of the experiment, subjects were required to listen to a pre-recorded tape containing spoken letters of the alphabet combined with a background of white noise. They were requested to record each letter as it was spoken and guess on the occasions where they were unsure. The results were presented as a 26x26 confusion matrix. Conrad reported that the errors which occurred in the visual task

occurred within both groups of letters. He also observed that in the verbal task, misheard letters were invariably replaced by ones which *sounded* similar to the correct letter. Significantly, these errors were similar to those produced in the visual experiment.

Subsequent to Conrad's (1964) finding that when stimuli are presented visually, errors occur between acoustically similar items, Conrad and Hull (1964) demonstrated that acoustic confusability had a significant influence over memory span. Six-letter sequences were selected from four vocabularies composing of either three or nine items, each either acoustically confusable or nonconfusable within-group. Items were presented visually and subjects were required to write down their recall attempt immediately after presentation of each sequence. Once again, the vocabulary from which the items were drawn remained in sight during recall.

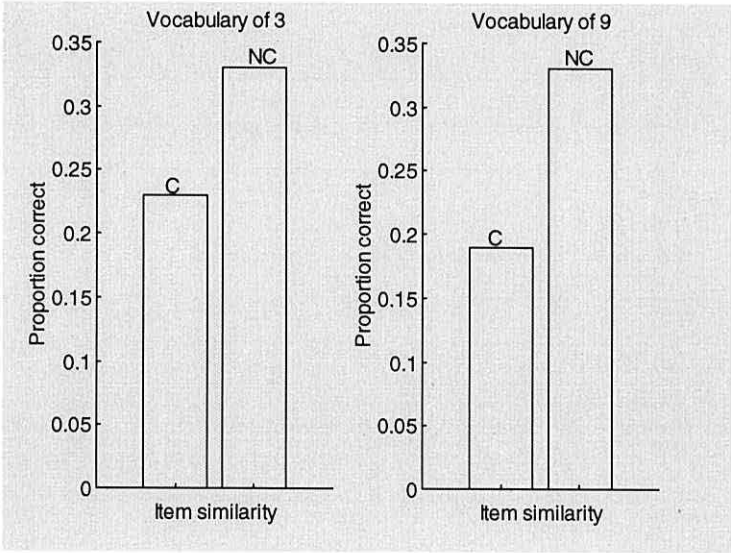

Figure 2.5 *Immediate recall of confusable (C) and nonconfusable (NC) six-letter sequences (Adapted from Conrad and Hull, 1964)*

Conrad and Hull scored one letter wrong if it was the incorrect letter for that serial position (i.e. a paired-transposition scored as two, a whole sequence incorrect, as six). They found that memory span was lower for the vocabularies containing acoustically confusable letters as is illustrated by figure 2.5. Conrad and Hull's findings illustrate that where the size of the vocabulary (i.e. hence information per item) is held constant, then memory span is affected by the probability of confusion *within* the vocabulary set.

More errors occur when the vocabulary contains letters that are acoustically confusable. This result also illustrates that the size of the vocabulary of items from which the stimuli are selected has little influence over memory span, consistent with the results of Drewnowski (1980).

Both Conrad (1964) and Conrad and Hull (1964) found a phonemic similarity effect with letter stimuli. Baddeley (1966, experiment 1, *acoustic similarity* condition) in an investigation of acoustic, semantic and formal similarity for verbally presented three-letter words (e.g. *man*, *mat*, *cat*, *can*, etc..) reported a similar effect (figure 2.6). He demonstrated that the effect of acoustic similarity was considerably larger than the effect of semantic similarity.

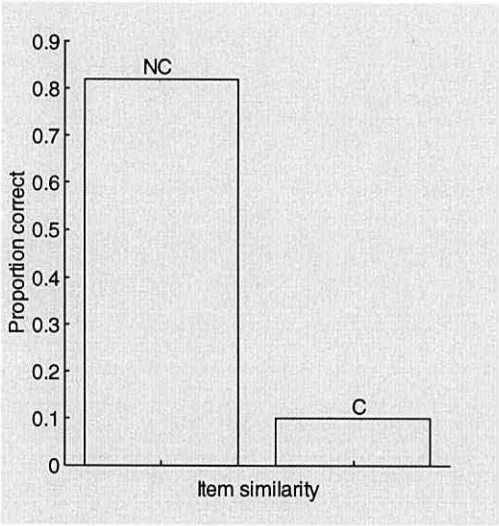

Figure 2.6 *Affect of acoustic similarity on memory span for words (Adapted from Baddeley, 1966)*

In an experiment in which subjects were presented with eight items, including both letters and digits, Wickelgren (1965b) reported that intrusion errors in short-term recall tended to have a vowel phoneme in common with the correct letter or digit. He observed that fewer errors occurred with *digit* stimuli than with letter stimuli, presumably as digits are more acoustically distinctive. Wickelgren (1965a) notes that acoustic similarity leads to failure to reproduce items in the correct *order* rather than failure to reproduce the items themselves.

In summary, phonemic or acoustic similarity between stimuli degrades serial order recall (Conrad, 1964; Conrad & Hull, 1964), increasing the number of order errors (Wickelgren, 1965a). This represents fundamental data that all formal models of serial order must address and must be taken into consideration when deciding how best to represent stimuli computationally.

2.5 Baddeley's phonemic similarity effect

In a series of experiments aimed at identifying why acoustically confusable items are more difficult to recall in order than nonconfusable items, Baddeley (1968) demonstrated that confusable and nonconfusable items are forgotten at similar rates. He inferred that order errors are not therefore attributable to an item storage process. However, in further experiments, Baddeley investigated whether these errors occurred during the retrieval stage and considered two hypotheses which made very different predictions about how performance would be affected when subjects were presented with a list of alternately confusable and nonconfusable items.

The first hypothesis was based upon Wickelgren's (1965a) inter-item association theory[1] in which each list item serves as the stimulus, or cue, for the next item. Wickelgren suggests that acoustically confusable items, with phonemes in common, are akin to repeated items in a list (Wickelgren, 1965c). As such, an acoustically confusable item in a specific serial position will provide a similar cue not only for the next list item, but also any other items that follow other acoustically confusable items in the list. This hypothesis predicts that subjects will recall acoustically confusable items in the wrong serial position, as *order errors*, and as a result they are likely to lose their place in the retrieval sequence. Therefore, in a list of alternately confusable and nonconfusable items, Wickelgren's hypothesis predicts that errors follow the similar items, and therefore that the nonconfusable items suffer the most during recall.

[1] Wickelgren's (1965a) *chaining based model* of short term memory is discussed in detail in the following chapter.

The second, Baddeley's "address hypothesis", suggests that the problem during recall is not one of the subject becoming confused over their location in the sequence during recall, but instead confusion as to which item to select during retrieval. If the items are more similar, so their retrieval cues become increasingly indistinguishable and hence it becomes harder to select the correct item given any one cue. When recalling a list of alternately confusable and nonconfusable items, this hypothesis predicts that the confusable items will suffer the most.

Subjects were presented visually with sequences of letters, drawn from two different sets: a pool of acoustically confusable (or similar, represented by an *S*) letters or a pool of acoustically nonconfusable (or dissimilar, represented by a *D*) letters. Six different types of sequence were generated (table 2.1): type A, alternately dissimilar then similar; type B, alternately similar then dissimilar; type C, a triple of similar followed by a dissimilar triple; type D, a triple of dissimilar followed by a similar triple; type E, six dissimilar items; type F, six similar items. No sequence contained a repeated item and the vocabulary of all twelve items was available to the subject throughout recall in an attempt to eliminate item errors.

Table 2.1

Six letter arrangements used during phonemic similarity effect (Adapted from Baddeley, 1968)

Type	Arrangement	Example
A	DSDSDS	JCWPLD
B	SDSDSD	CJPWDL
C	SSSDDD	CPDJWL
D	DDDSSS	JWLCPD
E	DDDDDD	JWLYRK
F	SSSSSS	CPDVBT

Recall was immediate and performance recorded in terms of the proportion of errors produced in each serial position (where an error was the failure to reproduce the correct item in the target serial position). The results for this experiment are reproduced in figure 2.7, however, here they are in terms of the proportion of items recalled *correctly* (i.e. 1-(proportion of errors)).

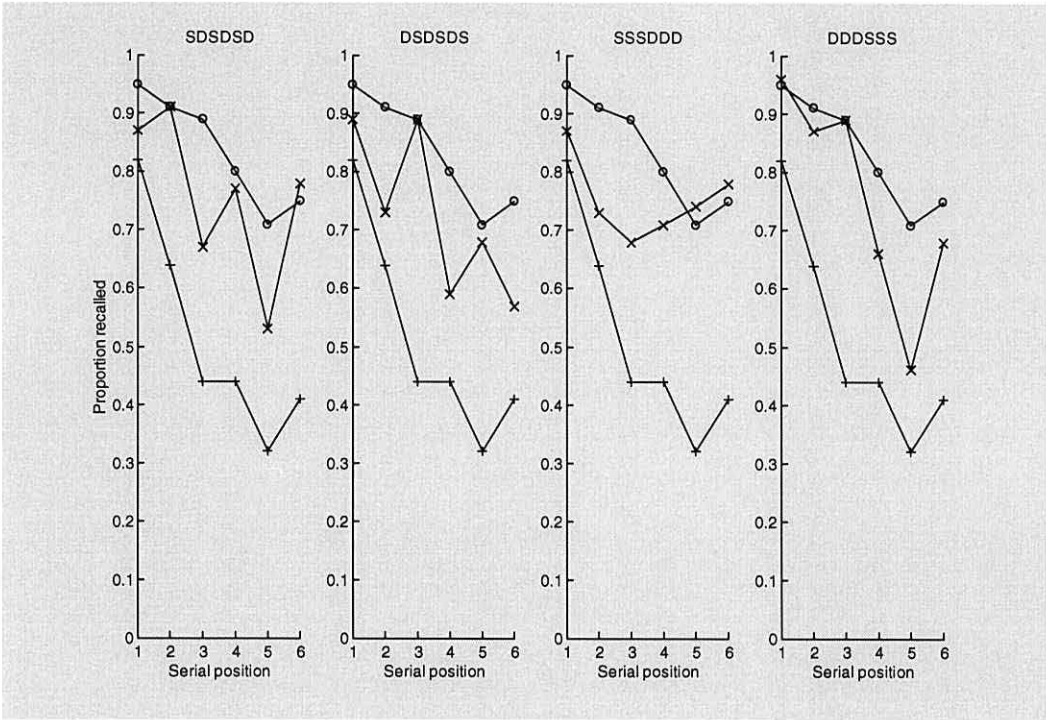

Figure 2.7 *Phonemic similarity effect for alternating lists*
(Adapted from Baddeley, 1968)

Examination of the mean proportion of recalls for each condition reveals that there is a clear phonemic similarity effect in the E and F conditions (mean proportion of errors: E=18.1%, F=49.2%) whereas there is very little difference in performance in the other four conditions where approximately 26% errors occur.

However, the serial position curves produced for each condition allow a clear conclusion to be drawn with respect to the hypotheses being considered. It is clear that the alternating list conditions (A, B, C and D) are bounded by the two pure conditions (E and F). In the alternating list conditions, dissimilar items are unaffected by the presence of similar items. It is clear, however, that than in each of the four conditions, errors occur on the acoustically confusable or similar items. This led Baddeley to reject Wickelgren's chaining based hypothesis.

In a replication of this experiment, Henson, Norris, Page and Baddeley (1996) observe that nonconfusable items are recalled better in the alternating conditions than in the pure condition, a result not evident in Baddeley's data, which Henson attributes to the

predictability of the consonants used in each list condition (Baddeley, Conrad & Hull, 1965).

In summary, in section 2.4 we reviewed evidence that acoustically confusable items are harder to recall accurately than acoustically nonconfusable items (e.g. Conrad & Hull, 1964). In the current section, we suggest that retrieval may be the source of these errors (Baddeley, 1968). In the following section, a more thorough consideration of order errors in a study by Henson, Norris, Page and Baddeley (1996), reveals that more order errors occur between items in the acoustically similar group (Wickelgren, 1965a) than in the acoustically dissimilar group. Baddeley's (1968, experiment 5) phonemic similarity effect has proved impossible for many formal models to replicate and as such, has become a benchmark by which to judge new models of serial order.

2.6 Transposition gradients

Serial position curves illustrate how the ability to recall the correct item in the correct serial position varies across list position. However, they do not reveal how errors are distributed across each serial positions (i.e. the distribution of order errors).

For example, if a subject is presented with the list *6BQ327* and they recall it as *6DQ237*, both an order and item error have been committed. Where *B* has been recalled erroneously as *D*, an item error has occurred. While, an order error occurs where the *3* and the *2* have transposed (Murdock, 1974).

It is possible to record the proportion of times that each item is recalled in each list position in order to generate a square array illustrating the distribution of order errors across a list (e.g. Fuchs, 1969; Estes, 1972; Henson, Norris, Page & Baddeley, 1996). Known as position functions, transposition gradients or distance functions, graphing these matrices allows a thorough analysis of order errors in serial recall tasks.

In this section, three examples of distance functions are presented: the first (Healy, 1974) illustrate the most basic, symmetric distribution for a four item list (Estes,

1972); the second (Fuchs, 1969) illustrates the effect that repeated presentation of a list of five words in a probe test for recall, has on the distribution of order errors; and the third reconsiders Baddeley's experiment 5 (1968) and examines the distribution of order errors occurring during an alternating list condition (Henson, Norris, Page & Baddeley, 1996).

In an attempt to address the differences in the shape of the serial position curves generated as a result of a two-part experiment separating item and order information, Healy (1974) considers the distance function for a four item list. These curves, generated as a result of her *order only* experimental condition (explained in section 2.7) illustrate how each item is recalled in various proportions in each serial position.

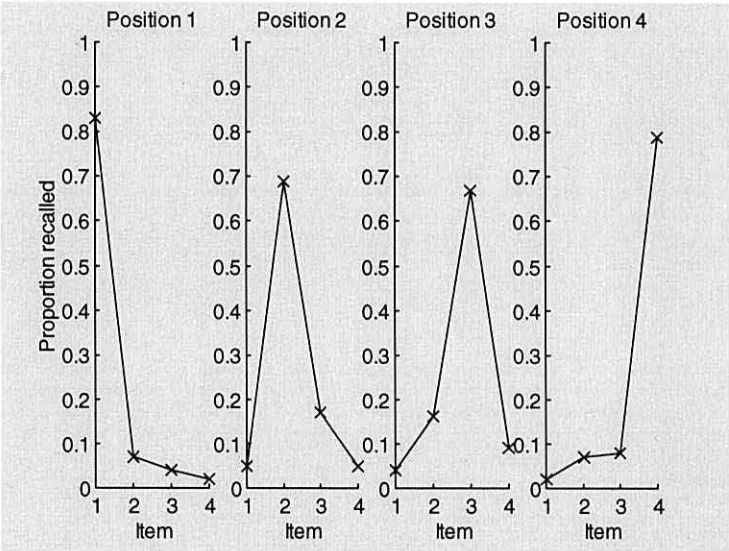

Figure 2.8 *Distance functions for the order only experiment (Adapted from Healy, 1974, three digit delay condition)*

Figure 2.8 reveals that each item is recalled most often in its target serial position and that the number of transpositions that occur decreases as the distance from the target serial position increases. This is most evident in the first list position where the first item is recalled the most, then each remaining item in smaller proportions than the item occupying the previous serial position. The figure also reveals that more transposition errors occur in the central two serial positions than in the outer positions. This

confirms Estes' (1972) hypothesis[2] which states that the likelihood of two letters being transposed depends on the separation between them. Furthermore, Estes observes that the distance functions for the first two positions are symmetrical, almost mirror images of those for the last two positions, except for the slight differences due to primacy and recency.

Fuchs (1969) in a probe recall task examining item and order errors in word recall, considers the effect of repeated presentation of stimuli on the form of the distance function. The results (figure 2.9) illustrate how each transposition gradient becomes more distinct, as the number of order errors decrease, after each presentation. For example, consider the third serial position in the single presentation condition (top row) with the corresponding position in the three presentations condition (bottom row). Clearly erroneous recalls involving the first and last item benefit only slightly from the repeated presentation of each list. However, the proportion of errors that involve the second and fourth item decrease with increased repetitions. Recall of the target item improves by 50% after the third repetition.

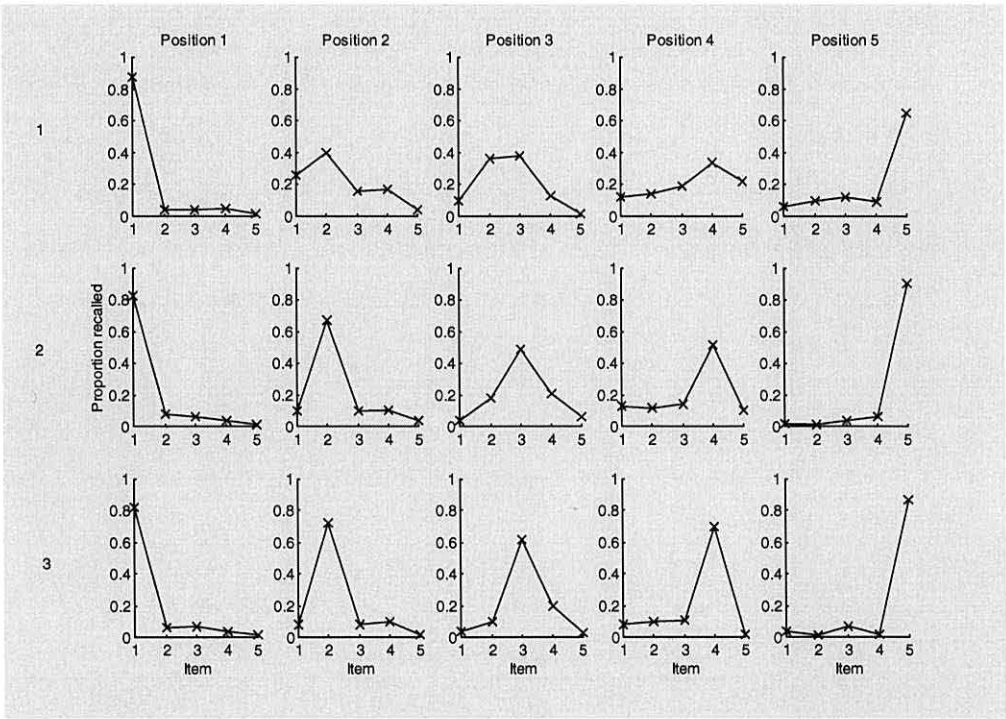

Figure 2.9 *Distance functions for a five item list after 1,2 or 3 repetitions (Adapted from Fuchs, 1969)*

[2] Estes' (1972) *perturbation model* of short term memory is discussed in the following chapter.

More precisely, the proportion of target responses increases, while the number of order errors decreases, with repeated presentation. Fuchs concludes that the order information for items in the centre of the list is learned more slowly than that same information for the items at the outer edges of the list.

A thorough analysis of error distributions produced during the phonemic similarity effect is provided by Henson, Norris, Page and Baddeley (1996). In a series of experiments replicating Baddeley's experiment 5 (1968), they present error distributions for the alternating list conditions that illustrate how transpositions occur between confusable items (i.e. confusable items are replaced by non-target confusable items; e.g. Conrad, 1965).

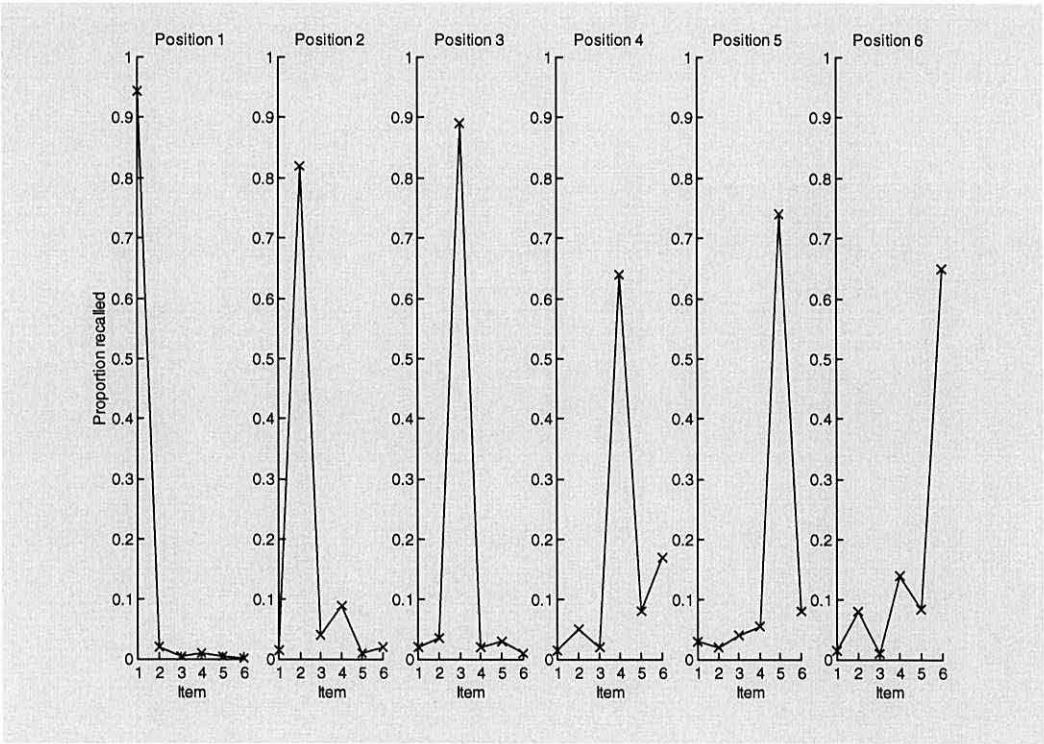

Figure 2.10 *Distance function for nonconfusable-confusable list*
(Adapted from Henson, Norris, Page & Baddeley, 1996)

This is clearly evident in figure 2.10 which illustrates the transposition matrix for the type A list (nonconfusable followed by confusable item) condition. This figure reveals how the confusable items are recalled in higher proportions than the nonconfusable items in the list positions where confusable items were presented. For example, in the

fourth serial position, the fourth item is recalled the most. However, the second and sixth items are also recalled in large proportions (5% and 17% respectively). A similar effect is evident in the second and sixth serial positions.

Clearly analysis of the serial position curve is insufficient when considering serial ordered recall performance. Distance functions allow the order errors produced during recall to be analysed. Curves produced by Healy (1974) confirm Estes' (1972) observation that transpositions involve greater proportions of the items occupying the serial positions immediately adjacent to a target serial position than those occupying the serial positions more distant from the target position. Henson, Norris, Page and Baddeley (1996), replicating Baddeley's experiment (1968, experiment 5), confirm that in the alternating list conditions, transpositions occur within acoustically-like groups. Clearly, any formal model of serial order must be capable of reproducing the distribution of order errors described here.

2.7 Item and Order errors

In the previous section, distance functions were introduced as a means to analyse order errors. In the following section, order errors are considered alongside item errors, and data presented which leads to the conclusion that item and order errors are the result of different mechanisms (Healy, 1974).

Conrad (1965) suggested that order errors were the result of pairs of item errors. However, Conrad's account fails to explain why transposition errors occur in such large quantities in short-term memory (e.g. Wickelgren, 1965a) and why loss of order information is faster than loss of item information (Bjork & Healy, 1974).

In a series of experiments attempting to separate the processing of item and order information (also the effects of increasing the retention interval before recall), Healy (1974) confirmed that serial position curves for item and order retention are different. In the first of two experiments, the *order only* experiment, Healy (1974) presented

subjects with letter stimuli[3] in sequences four items long. During recall, subjects were required to complete four blank boxes with the letters they were presented with on that trial, in the same order that they were presented. In order to minimise the number of item errors, each subject was given details of the consonants that they would be presented with before the session. Healy observed that the serial position curve produced in the experiment was bowed, with a first-item primacy and last-item recency component. In the second, *item only*, experiment, Healy presented subjects (in the *all-different* context condition) with complete order information in order to eliminate the possibility of order error occurring during recall. The consonants for the four-letter strings were drawn from a vocabulary of 12 items. The vocabulary contained four subsets of (three) letters and during presentation, only consonants from a given subset appeared in a given serial position. In this manner, by presenting subjects with all the item information for each serial position, order errors could be eliminated. Healy observed that the curve produced in the item only experiment lacked the bowing present in the order only experiment. Item errors occur in similar proportions in each serial position. The serial position curves for both experiments are presented in figure 2.11 and illustrate that order errors occur in greater proportions than item errors.

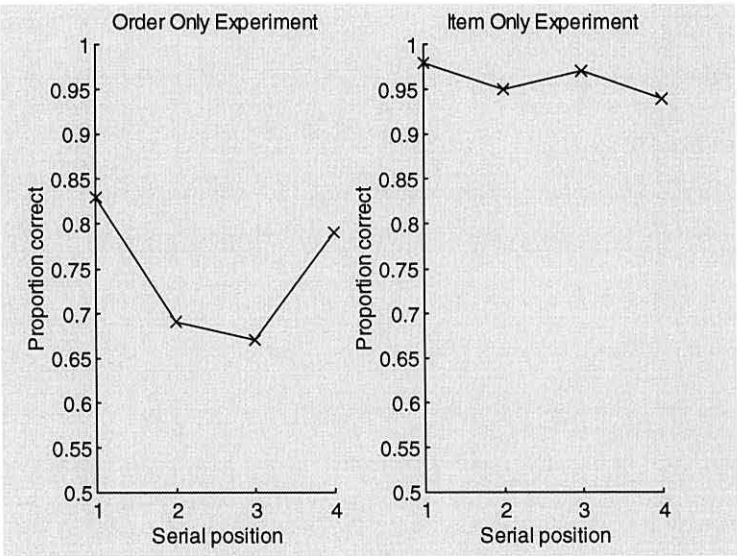

Figure 2.11 *Order only and item only data for four item list (Adapted from Healy, 1974, three digit delay condition)*

[4] The precise nature of the stimuli depends on which condition the subject is in: *All-different*, *Paired* or *All-same*. In the present discussion, we are concerned only with the *All-different* condition.

Analysis of the distance functions for the order only condition (figure 2.8), allows for rejection of Conrad's suggestion that order errors are in fact occurrences of item error couples. If this were the case, then the transposition errors revealed by the distance functions would be uniformly distributed across each serial position. Instead the transpositions are unevenly distributed about the target serial positions in accordance with Estes (1972) thus contradicting Conrad's hypothesis. Conrad's account also fails to explain why order information is lost more quickly than the item information (Bjork & Healy, 1974).

In summary, Healy (1974) demonstrated in a series of experiments that, contrary to Conrad (1965), item and order errors are the result of different mechanisms. Item errors occur uniformly across each serial position and in fewer proportions than order errors, which occur more rapidly and most often in central serial positions. A formal model of serial order must provide an associative mechanism capable of accounting for the differences in the serial position curves for item and order errors.

2.8 Repeated items and the Ranschburg Effect

The findings described so far have each used lists containing only unique items: at no point is the same item (letter, digit or word) presented to a subject in two consecutive serial positions. In this final section, we consider recall of lists that contain repeated items.

Wickelgren (1965c) presented subjects with sequences of between six and ten digits, at a constant rate of one digit per second (in a second experiment, this was increased to five digits per second). In the interval before the next sequence, subjects were requested to recall the previous sequence by filling-in the requisite number of blank spaces on an answer sheet. Subjects were allowed to guess or omit responses if they were unsure. Wickelgren tested each subject using twenty lists of digits for each of five list length conditions. Three lists contained no repeated items, the remainder contained at least one repeated item in one of a number of list positions. For example, one condition (iji_M) corresponded to a sequence containing one digit repeated,

separated by one digit in the middle (e.g. 154596). A second condition (ij--ij_{BE}) represented a sequence containing two digits repeated in pairs and occurring at opposite ends of the sequence (e.g. 760476).

Wickelgren analysed the data in terms of ordered and free recall for both individual items and sequences. He focused on errors that occurred after each repeated item, when both of the repeated items were recalled correctly. For example, for the sequence *ijikl...*, Wickelgren was concerned with the frequency with which k was replaced by j in the fourth serial position during recall, and j replaced by k in the second serial position. Wickelgren (1965c) referred to these as associative intrusion errors. Wickelgren reported that the items that followed repeated items were more likely to be transposed with each other than items following non-repeated items. This provided Wickelgren (1966) with evidence to reject non-associative models of serial order (e.g. Conrad, 1965) in favour of an associative, chaining based, hypotheses.

In Conrad's model, repeated items are maintained at more than one (fixed) location, so Wickelgren (1966) suggested that associative intrusion errors would therefore be counter intuitive in such a model. In contrast, he suggested, associative based models, where repeated items are only represented once but are connected by item-to-item associations, would permit errors of this form to occur.

Recall of repeated items is further complicated by the Ranschburg effect (Jahnke, 1969; Henson, in press). In order to demonstrate the Ranschburg effect, performance for a sequence of items (e.g. seven digits) containing no repeats is measured for the control condition. Subjects are then required to recall a similar list of items that this time contains a single repeated item, separated by a number of intervening items. The Ranschburg effect is defined as the poorer recall observed for the repeated items than was evident for the items in the corresponding serial positions in the control condition.

Clearly lists containing repeated items provide further complications during recall. A formal model of serial order should provide some mechanism capable of learning and recalling lists contained repeated stimuli.

2.9 Summary

In this chapter, a range of empirical findings for adult short term memory for serial order have been discussed. In the following chapter, a number of theories and models of short term memory, which attempt to account for one or more of these paradigms, are presented.

CHAPTER 3

Models and theories of memory for serial order

3.1 Introduction

The following chapter describes a number of different attempts at providing a formal model of the empirical data described in chapter 2. The models are classified by the manner with which they store serial order information for subsequent recall.

The first class of model stores items in fixed locations in the order in which they were perceived during presentation. Recall involves "reading" the contents of each location in order. These models are described in section 3.2 and are represented by Conrad's *bin* model (1965).

The second class of model, described in section 3.3, are the chaining models. First described by Ebbinghaus (1913), chaining models store order information in the links that connect one item to the next. Recall requires the first item to be recalled, then used as the recall cue for the next and so forth for each remaining item. More recently, chaining-based accounts (e.g. Lewandowsky & Murdock, 1989) have demonstrated the ability to reproduce a range of empirical results. Recurrent networks (Jordan, 1986; Elman, 1989) also employ the use of a chaining mechanism during learning.

Johnson (1970) and Estes (1972) both employ the use of a hierarchical structures to store learned sequences. Estes (1972) describes a model where item information is stored in a hierarchical structure while order information is preserved by a rehearsal process between each item and a control element. These hierarchical models are described in section 3.4.

The original network model of the articulatory loop (Burgess & Hitch, 1992) employed both item-to-item chaining and item-to-context association. Each of these models associates items with different states of a dynamic control signal. Item recall occurs when the learned-context is reinstated. Houghton's competitive queuing model (1990, 1994a) uses a two-dimensional control signal while Burgess and Hitch (1992; 1996) employ a high dimensionality context signal.

The final class of model to be introduced in this chapter, relies on an activation gradient across each learned item (e.g. Page & Norris, 1995). These models recall the most active item in each list position and are described in section 3.6.

3.2 Bin models

Conrad (1965) describes a non-associative (Wickelgren, 1966) model of memory that uses positional cues (fixed locations that Conrad referred to as *bins*) with which to temporally locate each stimulus item.

Conrad (1965) presented order error data generated during an immediate recall task for sequences of letter stimuli. Analysis confirmed that when errors occurred, items were likely to be replaced by an acoustically similar letter (Conrad, 1964). Furthermore, Conrad suggests that these same letters tended to transpose when presented together in a sequence. He proposed that order errors could in fact be the result of pairs of item errors. In an attempt to address this, Conrad outlined a non-associative *bin* model of serial order memory.

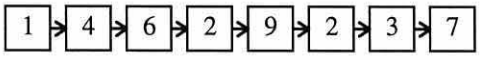

Figure 3.1 *Non-associative theory of memory for a sequence of digits (e.g. Conrad, 1965)*

Conrad suggests that each item perceived during learning is stored in a container whose location is fixed and corresponds to the to-be-learned items serial position (figure 3.1). During recall, subjects simply read the contents of each container in the

order requested by the experimenter. Item errors occur when the contents of a container, which decay over time, fall below a noise threshold and become imperceptible above the background noise as a result.

Conrad uses the model in an attempt to account for the production of order errors. He suggests that during recall, when the signal corresponding to an item contained in one of the bins falls below the threshold and into the background noise, the perceptual process responsible for recall may be similar to that for perceiving letters against a background of noise. Conrad explains that when a subject fails to recall an item correctly, it is highly probable (particularly if the items are acoustically similar) that the response for the next bin will be selected. Therefore, when attempting to recall the contents of the next bin, and if the subject rejects repetition for any reason, then it is likely that they will select the response that was correct for the previous serial position. A transposition error can therefore be attributed to item errors occurring between confusable items. It is also possible that when a signal falls below the noise threshold, an omission or guess may occur. This may involve the corresponding item from a previous sequence, resulting in a serial order intrusion error (Conrad, 1960a). Furthermore, Conrad describes a separate dynamic *availability* store of responses which subjects use to compare with the contents of the bins during recall. It is apparent how the model would not be affected by the size of item vocabulary (Conrad & Hull, 1964).

Wickelgren (1966) applies a non-associative model of memory, similar to Conrad's bin model, to the problem of accounting for associative intrusion errors. Wickelgren had reported (Wickelgren, 1965c) that when subjects attempted to recall a list containing a repeated item, even when they recalled the repeated items correctly (and in the correct serial positions) there was a chance that subjects would transpose both of the items following the repeated items. If a to-be-learned list contains a repeated item, both representations of the item are stored in the model (e.g. figure 3.1). Wickelgren (1966), making the assumption that storage or recall errors do not depend upon the contents of each bin, concluded that there is no reason for associative intrusion errors to occur in a non-associative model.

Conrad's account also fails to explain why transposition errors occur in such large quantities in short-term memory (e.g. Wickelgren, 1965a) and why loss of order information is faster than loss of item information (Bjork & Healy, 1970).

In summary, Conrad (1965) describes a model of serial order in which items are stored in order in fixed locations called bins. The contents of each bin decay with time. This introduces item errors (Conrad, 1964) and serial intrusion errors (Conrad, 1960a). Conrad also suggests that order errors result from pairs of item errors occurring in the same list. However, if this were the case, it would contradict the findings that item and order errors appear to be the result of different mechanisms (Bjork & Healy, 1970; Healy, 1974). Wickelgren (1966) observes that as errors occur at each bin, regardless of contents, associative intrusion errors occurring after repeated items are unlikely. Also, it is unclear how the bins come to be searched in the correct order during recall.

3.3 Chaining models

3.3.1 Basic chaining model

One of the first theories for serial order recall was presented by Ebbinghaus (1913). Ebbinghaus suggested that items are stored in memory as a sequence of associations linking the internal representation of one item with the next, in the same order in which they were presented. The strength of the association between any one item and the next depends upon the displacement between the two items. The association is strongest between adjacent items. Associations are directional and *point* towards the next item in the sequence. Accurate serial recall depends upon the subject being able to recall the first item in the sequence, and then use it as a recall cue for the next item (e.g. Wickelgren, 1965a).

For example, for the list of items, *abcd*, the strength of the association between *a* and *b* is greater than the strength of the association between *a* and *c*, while the association between *b* and *c* is greater than that between *b* and *d*. The weakest association is between *a* and *d*. Ebbinghaus proposed that, during recall, if item *a* was recalled, then

the strongest forward association connected to *a* would be to item *b*. Therefore, the second item could be retrieved by presenting *a* as a cue, and recalling *b* in the next position. Once *b* has been recalled, so the strongest association connected to item *b* will be to the third item, *c*. By ensuring that only *unidirectional* associations are employed, recall can be maintained in one direction only. Therefore, in this manner each of the list items may be recalled in the order in which they were presented.

Wickelgren (1966) examined the ability of a chaining, or *associative*, based model to account for intrusions in immediate recall of lists of digits containing repeated items. Figure 3.2 illustrates the list *14629237* and contains the repeated item *2*. Each internal item representation is connected with the next in the sequence by a forward association. Also, note that although the item *2* is presented twice during learning, it only has one internal representation in a chaining model of the sequence. Wickelgren suggests that subjects will recall the early portion of the list correctly as *1462*, before having to decide which item to recall next when presented with the first occurrence of the repeated item as a cue.

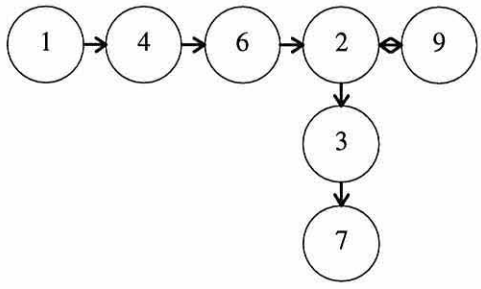

Figure 3.2 *Associative theory of memory for a sequence of digits*

Ideally, subjects will recall item *9* as the next in the sequence, and because of the *bi-directional* association connecting the *9* to the *2*, will recall the second occurrence of the stimulus item, *2*, next. The subject should now complete the sequence by recalling *3* and *7*. Wickelgren suggested that if item and order information were stored in this manner, associative intrusion errors occurring in sequences containing repeated items could be accounted for by a chaining based model containing single representations of each item interconnected by directional associative connections.

Wickelgren (1969) also outlines a structure for chaining that uses phonemic representation and includes order information by subscripting the current phoneme with information about the phonemes either side of it. These are referred to as wickelphones (Rumelhart & McClelland, 1986). For example, the word *EVERY* would be represented in typical chaining parlance as *E-V-E-R-Y*, however, in context sensitive chaining using wickelphone notation this would become ${}_{\$}E_{v}\ {}_{e}V_{e}\ {}_{v}E_{r}\ {}_{e}R_{y}\ {}_{r}Y_{\$}$. Here the *$* corresponds to a start and end marker.

Evidence against chaining based models is presented by Baddeley (1968, experiment 5). Baddeley describes an experiment aimed at identifying whether a retrieval mechanism is responsible for the finding that acoustically confusable items are harder to recall than acoustically non-confusable items (Conrad & Hull, 1964). Wickelgren's (1965a) inter-item association theory would suggest that acoustically confusable items would act as poor recall cues and as a result, performance for the non-confusable items relying on the confusable cues, would suffer (Baddeley, 1968). In fact, the converse is found, which suggests that a chaining based model of serial order will be unable to account for Baddeley's (1968, experiment 5) phonemic similarity effect. Also, Lashley (1951) criticises item-to-item based models for their inability to account for the complexities of rule-based behaviour such as language generation and also the fact that the majority of serial order tasks (e.g. fast motor control such as that used to move fingers when playing the piano) require a swifter execution than is capable of a simple chaining account.

In summary, a chaining based account of short-term memory is based upon internal representations of stimuli, interconnected by directional associations. Recall relies on items being presented as cues in order to recall the next item in the sequence. Repeated items are only represented in memory once, but are connected to neighbouring items in the sequence by bi-directional associations (Ebbinghaus, 1913; Wickelgren, 1965a). Chaining based models offer an explanation for associative intrusion errors that occur immediately after repeated items (Wickelgren, 1966) but fail to predict the phonemic similarity effect (Baddeley, 1968) and the complexity and efficiency of typical serial ordered behaviour (Lashley, 1951).

3.3.2 CHARM

Metcalfe Eich (1982) introduces one of the first distributed models of memory that uses the mathematical processes of convolution and correlation for item storage and retrieval. CHARM (Composite Holographic Associative Recall Model), is based on the concept of holographic association[5]. We discuss Metcalfe's model as convolution and correlation are important as the basic associative method that underlies a subsequent model of serial order.

Items, or patterns of features, are represented in CHARM as multidimensional vectors with elements whose values are continuous (taking both positive and negative values) and have an expected mean of zero. Item features are ordered which means that the fifteenth feature of one vector represents the same attribute as the fifteenth feature in a second vector. In this manner, related items can be created by duplicating features for each of the items. The similarity between items can be measured by taking the dot product between each item. Furthermore, by normalising the items, Metcalfe Eich ensures that the dot product of any item with itself is one - a measure that one item is identical with the other. If the dot product of one item with an other is zero, the items can be said to be unique and completely unrelated to one another (see Goebel & Lewandowsky, 1991).

Briefly, during learning CHARM performs accurate single trial learning storing the convolution of novel pairs of items in an episodic memory. The trace vector produced as a result, is of a larger dimensionality than the item vectors and bears no resemblance to any of the features in the stimuli. During retrieval, a cue is presented and correlated with the memory trace. If the stimuli are sufficiently distinct, then an approximation to the second of the stimuli can be recovered from the memory trace. However, if the stimuli are less distinct, a weaker approximation will be generated, blurred by the effects of noise and confusion between the stimuli. The retrieved item may then be deblurred by comparing it with the lexicon of recallable items and selecting the item it

[5] See Metcalfe & Murdock (1981), Metcalfe Eich (1982) and Murdock (1982) for more details.

bears most resemblance to by computing the dot products. This basic architecture is presented in figure 3.3.

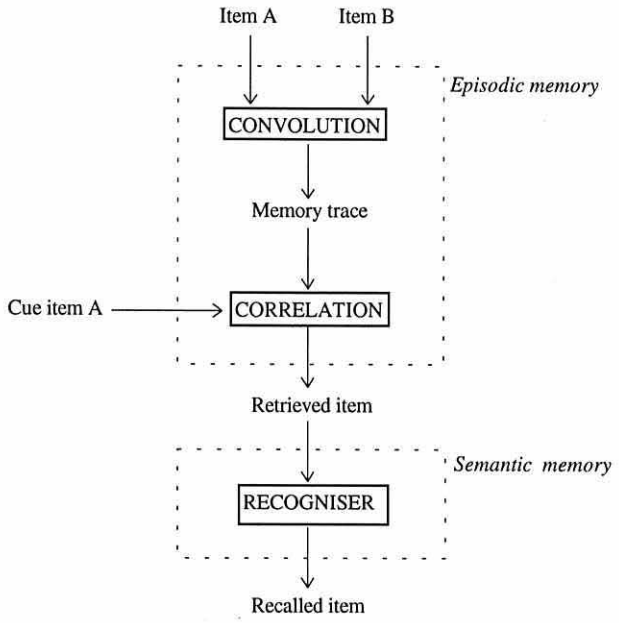

Figure 3.3 *CHARM's method of item encoding, decoding and recognition*

However, thus far CHARM would only have the capacity to learn a single association. In order to increase the capacity for learning, CHARM uses a *composite* memory trace in which it accumulates, feature by feature, each memory trace as it is generated by each new pair of stimuli. The composite memory trace is analogous to a photograph that has been exposed repeatedly - the photograph will contain ghost images of all of the previous exposures.

Table 3.1

Composition of the memory trace vector after presentation of the jth pair of items

j	Pair	Composite trace vector
1	A-B	**A*B**
2	C-D	**A*B + C*D**
3	E-F	**A*B + C*D + E*F**

As the first pair of items (**A** and **B**) is presented to CHARM, so the convolution (**A*B**) is stored in the composite memory trace. As the next pair of items is presented and the convolution calculated, so this too is combined with the current contents of

the memory trace. This process continues for each new pair of stimuli and is summarised in table 3.1.

At recall, it is the composite trace that is correlated with each individual cue item. If each of the items stored in the composite trace are unrelated, then using a composite memory trace during recall has the effect of adding noise to the retrieved item. The composite trace introduces interference based forgetting. Metcalfe and Murdock (1981, figure 1) demonstrate that as the number of associations stored in the composite trace increases, so performance degrades rapidly. This interference is even greater when there are a number of similar items in the list of to-be-learned items.

Ambiguity at recall is minimised by the inclusion of the pattern recogniser component of the model which permits redintegration of stimuli. The pattern recogniser contains two parts: the first is the lexicon of items presented during learning which serves as a limited vocabulary of possible responses; and the second is a matching process, in this case implemented using the dot product between the retrieved item and each of the lexicon items.

In summary, Metcalfe Eich (1982) outlines an architecture for association capable of single trial learning. Association is by convolution and recall by correlation. The memory trace generated during each presentation is accumulated in the composite memory trace which introduces interference based forgetting. CHARM, during a paired associate recall task, replicates the acoustic similarity effect (cf. Conrad & Hull, 1964) has also been shown to replicate the gradual-unlearning A-B A-D paradigm (Lewandowsky, 1991). However, in the next section we present a model of serial order that develops this associative mechanism.

3.3.3 TODAM

Metcalfe Eich (1982) applied a convolution and correlation distributed memory to the problem of paired associate learning. Murdock (1982, 1983, 1992, 1993) presents a general theory of distributed associative memory, TODAM, which stores item and

order[6] information in a distributed composite memory trace. Like CHARM it represents items as vectors of features and uses convolution and correlation as the associative mechanisms.

Like with CHARM (Metcalfe Eich, 1982), items are represented as vectors whose features take random values. Item and associative information is stored in the composite memory trace. Serial order information is derived from the associative information stored during each learning iteration. TODAM employs a chaining mechanism similar to that described by Wickelgren (1965a) where each item acts as the cue for the next. This may be modelled by *overlapping* successive associations: the first item is associated with the second, the second with the third, etc., for each item in the list. This process may be expressed mathematically as:

$$\mathbf{M}_j = \alpha \mathbf{M}_{j-1} + \gamma \mathbf{f}_j + \varpi \left(\mathbf{f}_j * \mathbf{f}_{j-1} \right) \tag{3.1}$$

Here, $\mathbf{f}_j$ and $\mathbf{f}_{j-1}$ are two items, and $\mathbf{f}_j * \mathbf{f}_{j-1}$ the convolution between the two items. $\mathbf{M}_{j-1}$ represents the contents of the memory trace prior to this iteration. α represents the forgetting parameter, while γ and ω represents the weighting parameters for the item and order information respectively. Information is therefore lost through both interference (through the use of a composite memory trace) and also decay (through the use of a forgetting parameter).

As with Metcalfe Eich's (1982) CHARM model of association, convolution is used to associate the current stimulus item with the previous. As before, for n-dimensionality stimulus vectors, the vector resulting from the convolution will have dimensionality $2n$-1 elements which means that the individual features of each stimulus item will not correspond to features of the memory trace. Also, before each item is added to the larger composite trace vector, it is padded with zeros in the n-1 elements either side of the centralised item elements. The contents of the composite memory trace after the presentation of the jth item are illustrated in table 3.2.

[6] In contrast to CHARM where only order information is stored in the associations *between* pairs of simuli.

Table 3.2

Composition of the memory vector, M_j, after presentation of the jth item

j	Item	$\mathbf{M}_j$
1	A	$\mathbf{A}$
2	B	$\gamma\mathbf{B} + \omega(\mathbf{B}*\mathbf{A}) + \alpha\mathbf{A}$
3	C	$\gamma\mathbf{C} + \omega(\mathbf{C}*\mathbf{B}) + \alpha(\gamma\mathbf{B} + \omega(\mathbf{B}*\mathbf{A})) + \alpha^2\mathbf{A}$

Item recall involves correlating the first item with the memory vector in order to generate an approximation to the second item. Item recognition relies on taking the dot product between the probe item (in this case, the approximation to the second item) and the memory vector. The decision as to whether or not the probe item was learned is taken on the basis of whether or not the dot product exceeds a threshold value.

Serial-order recognition is instigated by reproducing the memory vector after each item has been recalled and comparing it with the original trace at the corresponding stage during learning by taking the dot product. In a similar fashion to before, whether or not the two are similar depends upon the dot product. This mechanism is significant as it allows TODAM to accurately recall the item information for a simple list, *ABC*, but to confuse the order information, recalling the list as *ACB*.

TODAM has been applied to a range of serial-order data (Lewandowsky & Murdock, 1989; Baddeley, Papagno & Norris, 1991; Murdock, 1993) with varying degrees of success. Lewandowsky and Murdock (1989) attempt to fit a refined version of TODAM to a number of serial order paradigms including serial learning, memory span, partial report effects, delayed recall, list length effects, hebb repetition effects, similarity, build up and release from PI, the primacy and recency dissociation, forward and backward recall and positional probe effects.

Lewandowsky and Murdock (1989) subscript the item and order information weighting parameters so that they can vary across serial position. Furthermore, Lewandowsky and Murdock suggest that when an item ($\mathbf{f}_{j-1}$) is presented as a cue for the next item ($\mathbf{f}_j$) in the sequence during recall, if the cue item has itself been recalled inaccurately, an approximation to it will be sufficient for accurate probing:

$$\left\{\begin{array}{ll} \mathbf{f}_{j-1}\#\mathbf{M} = \mathbf{f}'_{j-1} & \text{if retrieval j - 1 successful;} \\ \mathbf{f}'_{j-1}\#\mathbf{M} = \mathbf{f}'_{j} & \text{otherwise.} \end{array}\right\} \tag{3.2}$$

Significantly, Lewandowsky and Murdock suggest that, as defined, TODAM will not learn, as repeated presentation of the same items will not lead to better performance. Therefore, they suggest the following modification to the current *open loop* architecture: the addition of a feedback-loop which determines the amount of new item and order information provided by the jth item. However, the development of this *closed-loop* architecture is at the expense of computational complexity and results in an inability to compute the probability of recall (Lewandowsky & Murdock, 1989, p. 31).

When implementing an open loop simulation, sequences are bounded by start and end signals (cf. Shiffrin and Cook, 1978). The order information parameter, ω, decays exponentially with serial position, while γ changes in complement to ω (i.e. associative information is learned most strongly at the beginning of a sequence, while item information is learned most strongly at the end of a sequence). Furthermore, when an item is retrieved successfully, the retrieved item is *added* to the composite trace before it is presented to the composite trace as a probe for the next item.

A series of alternate retrieval mechanisms are outlined including one which uses an *anticipation* paradigm, where a deblurred version of the previous item is always available as a probe, and a second, which allows the use of the retrieved approximation as a recall probe (equation 3.2).

Implementing the closed loop model, sequences are bounded in a similar fashion to the open loop version. Each feature of the composite trace is reset to zero before training can proceed. Once again, ω and γ vary exponentially across serial position, although in this case apriori knowledge of the list length is expected (Lewandowsky & Murdock, 1989, equation 7). Retrieval assumes that a deblurred item will be used as each recall probe. The closed loop architecture requires knowledge of which items are available for recall. As subjects seldom repeat an item they have already recalled, competitors

are selected from the target items following the probe (Lewandowsky & Murdock, 1989, p. 35). Therefore, in the sixth serial position of a nine item list, competitors include only the sixth, seventh, eighth and ninth items. An additional *N* items are also available as competitors (although *N* is typically set to only zero or one).

Finally, before considering how TODAM is fit to the empirical data, both the open and closed loop models share a number of fixed and free parameters. Fixed parameters, which aim to be constant for each simulation. include the size of the memory vector and the recall tolerance limits. Free parameters, which may alter for each simulation, include the forgetting parameter, the item and order weighting parameters and the number of competitors available during recall.

Briefly, both the open loop and closed loop TODAM architectures are applied to the modelling of a number of serial order benchmark paradigms. In order to fit the open loop model using the anticipation procedure to the empirical data, a further free parameter is introduced which allows the rate constant for the exponentially decreasing associative weighting parameter, to decrease exponentially itself.

The open loop implementation of TODAM is applied to the problem of modelling the serial position curve. The fit is adequate, with slightly less recency than the empirical data. However the trough of each curve moves from the beginning of the list towards the end as learning increases in contrast to the empirical data where it is fixed. Applying the closed loop version of the model reveals a closer approximation to the empirical data (Lewandowsky & Murdock, 1989, figure 9). Attempting to fit the memory span data of Crannell and Parrish (1957) with the open loop model, the forgetting parameter α is varied in order to simulate the different materials learned during the empirical data. Once more an adequate fit is provided (Lewandowsky & Murdock, 1989, figure 12). Furthermore, the open loop model provides a fit to a partial report paradigm and also the delayed recall task (Lewandowsky & Murdock, 1989, figures 13 & 14).

Although Murdock (1992, 1993) attempts to refine TODAM in order to account for chunking and paired-associate effects, TODAM is fundamentally a chaining based model of serial order, and is therefore susceptible to similar errors and deficiencies as the basic chaining model of Wickelgren (1965a). Baddeley, Papagno and Norris (1991) attempt to fit TODAM to Baddeley's (1968, experiment 5) phonemic similarity effect data. To review, a chaining based account would predict that when each confusable item is presented as a recall probe, recall of the nonconfusable item following it will suffer. However, the empirical data illustrates that the converse is true, nonconfusable items do not suffer from the proximity of confusable items. Baddeley, Papagno and Norris (1991, figure 10.3) illustrate how, for the *DSDSDS* list condition, performance for both the confusable and nonconfusable items is almost identical to that for the nonconfusable only list condition. There is no "sawtooth" effect and performance for the confusable items is as good as that for the nonconfusable items. A similar effect is present in the *SDSDSD* condition.

Mewhort, Popham and James (1994) provide the first of two substantial critiques of TODAM. They argue against three of the core assumptions: the chaining mechanism, the availability of competitors during recall and also the complementary relationship between the item and order weighting parameters.

In addressing the first, Mewhort, Popham and James (1994) reject Lewandowsky and Murdock's (1989) suggestion that the use of a retrieved item as a cue will produce similar performance to that when using the correct probe item, regardless of whether or not it was recalled correctly (i.e. anticipation procedure). They observe that as soon as an error occurs, the cue item will be incorrect and as a result, recall will be prevented in the list positions following that containing the error (e.g. the chain will be broken). Lewandowsky and Murdock suggest that using either the retrieved, or facsimile, item (i.e. the item retrieved from the composite, before it is compared to the competitors and deblurred) or the correct cue item (e.g. the anticipation procedure) will produce similar performance during recall. However, Mewhort, Popham and James (1994) observe that using the facsimile item results in chance performance, the overall behaviour is dictated to by the free parameters, in particular the number of

competitors available during recall. This is confirmed by Lewandowsky and Li (1994) who demonstrate that under some conditions, cueing with the facsimile is more than adequate.

Secondly, Mewhort, Popham and James (1994) argue that all of the lexicon of items should be available as competitors during recall and not just the remaining items, as this artificially introduces a high degree of recency into the system.

Finally, Mewhort, Popham and James (1994) argue that TODAM is biased against items at the start of the list as the item weighting parameter is minimal at the beginning of a list. Furthermore, that as the item and order information weighting parameters complement each other, even when there is no item information, order information should be perfect. This suggests that perfect recall is possible even without any item information. Lewandowsky and Li (1994) counter this by highlighting that this only occurs for the *context cue*, recalled immediately before the first list item.

Nairne and Neath (1994) provide the second critique of TODAM. They reject the sampling without replacement method by which items are removed from the lexicon of competitor items during recall. They illustrate that if, once an item has been recalled, correctly or otherwise, that recalled item is removed from the lexicon of competitors, all recency in the serial position curve is destroyed. However, if, in accordance with Lewandowsky and Murdock (1989), the target item for that position, regardless of whether or not it was in fact recalled correctly, is removed from the lexicon, a huge recency effect is introduced. Specifically, if the item for position five in an eight item list is retrieved, it is compared with items six, seven and eight. Regardless of which item is recalled from that subset, it is item five that will be removed from the lexicon of competitors before recall proceeds to the next serial position. Clearly, this will favour recall of the last item as there will be no competition during recall. By introducing the ad hoc free parameter, N, Lewandowsky and Murdock can vary the degree of recency by adding competitors to the list. This also results in truncated transposition gradients lacking the symmetry of those reported elsewhere (e.g. Estes, 1972; Bjork & Healy, 1974; Henson, Norris, Page & Baddeley, 1996).

In response to the criticisms of the deblurring and sampling without replacement techniques employed by Lewandowsky and Murdock (1989), Lewandowsky and Li (1994) suggest the use of a *brain-state-in-a-box* in order to deblur retrieved items and the use of *anti-learning* with which to remove both associative and item information from the composite memory trace after recall. Employing both of these in a simulation, they report reasonable performance although the problem of asymmetric distance functions remains.

In summary, TODAM (Murdock, 1982, 1983, 1992, 1993; Lewandowsky & Murdock, 1989; Lewandowsky & Li, 1994) is a chaining based distributed model of serial order. Items, represented as vectors of features, are associated in pairs by convolution and stored alongside representations of the items themselves, in a composite memory trace vector. Item recall involves presenting a recall cue, typically the current list item, and comparing the retrieved item with the memory trace in order to ascertain whether that item was presented during learning. If it was, then it can be used to recall the next item in the sequence. If it was not presented, then the approximation may be sufficient as a recall probe. Recall of order information is possible by trying to regenerate the memory trace vector using the recalled items, and comparing it with the original memory trace at each step. Once an item is recalled, it is removed from the lexicon of competitor items available during recall. However, if an item is incorrectly recalled, the target item that should have been recalled may in fact be removed from the competitors list. TODAM, in both the open loop and closed loop or feedback version, has been applied to a number of serial order paradigms successfully. These paradigms include the serial position curve, similarity and memory span effects. However, like previous chaining models, TODAM is unable to replicate Baddeley's (1968, experiment 5) phonemic similarity effect (Baddeley, Papagno & Norris, 1991). Furthermore, a number of criticisms are raised at TODAM (Nairne & Neath, 1994; Mewhort, Popham & James, 1994) in particular the limitations of the competitors lexicon, asymmetric transposition gradients, arbitrary weighting of item and order information across serial position and the underlying weakness of a chaining mechanism for recall.

3.3.4 Recurrent networks

A second family of chaining based models are recurrent networks (Jordan, 1986; Elman, 1990; Chater & Conkey, 1994). These networks connect either the output units (Jordan, 1986) or the hidden layer units (Elman, 1986) to the input. This means that the network's output (i.e. current state) can become part of the input that acts as the cue for retrieval of the next item.

Jordan (1986) describes a connectionist network whose recurrent connections associate a static pattern, or plan, with a serial ordered input pattern. When an input is presented to the network, the signal propagates first to the hidden layer of units and then to the output units. Learning is by back-propagation (Rumelhart, Hinton & Williams, 1986). However, also presented to the hidden units are the outputs from the state units. These units correspond to the previous state of the output units. Therefore, the Jordan network performs a series of chaining associations between the most recent set of input items and the previous state of the output units. This architecture is presented in figure 3.4*a*. If such a network were required to learn and then recall the phoneme sequence /strIng/, it would recall them as the sequence, /s/, /t/, /r/, /I/ and finally, /ng/.

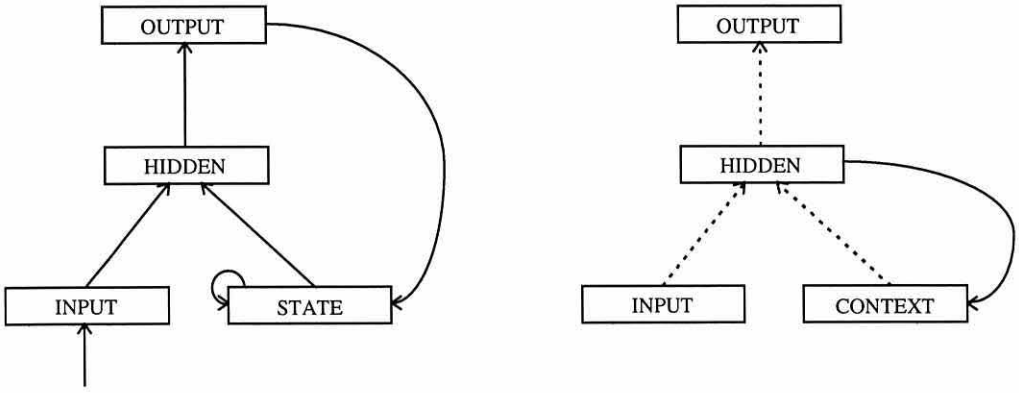

Figure 3.4 (**a**) *Jordon recurrent network (1986)* (**b**) *Elman recurrent network (1990)*

Elman (1990) refines the architecture by instead connecting the hidden layer of units to a hidden layer of context units via a set of fixed weights. Learned weights link the

input vector to the hidden layer, and also the hidden layer to the output layer of units. In this manner, every item presented to the network is associated with a context representing the state of the hidden units just prior to the item being presented. This architecture is presented in figure 3.4*b*. Elman (1990) applies the network to the problem of predicting the next letter in a sequence with some success even when the sequence contains repeated items.

However, recurrent networks, like other chaining-based accounts of serial order, will erroneously predict that items following confusable items will suffer because of the similarity between the retrieval cues. This applies even though each cue is some composite of each of the previous items. Also, recurrent networks do not possess the ability to recall each item in a sequence autonomously as the sequential structure is not self-generated. Furthermore, recurrent networks that learn sequences by back-propagation are clearly unsuited to modelling single trial learning (Houghton & Hartley, 1995).

3.4 Hierarchical models

3.4.1 Chunking model

Johnson (1970) develops a model based upon the concept of chunking in short term memory as first suggested by Miller (1956). Miller suggested that there is a limit on the number of chunks of information that can be held in immediate memory, although how much information (bits) those chunks may represent is not limited. For example, a memory span of five words could be described as a memory span of fifteen phonemes (given that each word contained, on average, three phonemes). Furthermore, subjects demonstrate that their ability to recall sequences of binary digits is improved if they recode the sequences into denary digits prior to learning (Miller, 1956).

Johnson defines codes and chunks as follows. A chunk is a sequence which is represented in memory by a single code. A code can represent both item and order information and is logically distinct from the information it represents. During recall, a

code must be recovered before the information it represents can be extracted - an "all or nothing" recall strategy.

If it is assumed that subjects chunk items in a hierarchical set during learning, then retrieval involves decoding the codes in the hierarchy into the information that they represent. Johnson's decoding operation assumes that in order for subjects to recall an ordered sequence, they must first recall a coding device that represents the entire sequence as a single chunk. Then the subject proceeds using a depth first search strategy through the hierarchy, decoding each of the codes into its components. Once the subject has expanded the very lowest level of that branch of the hierarchy, the subject extracts the next most recently stored component from short-term memory and proceeds to expand that branch of the hierarchy as before.

In the first step, the stimulus elicits an arbitrary code ('1') that represents the entire sequence (of nine letters, beginning with S, B and J) that the subject has just learned. The subject immediately decodes the stimulus into all of its immediate components, in the present case, the codes 'A', 'E' and 'U' that represent three chunks (figure 3.5).

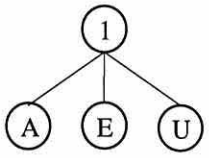

Figure 3.5 *Decoding the first memory code into the three codes representing the chunks A, E, U (Adapted from Johnson, 1970)*

Next, the most recent of the three is expanded whilst the remainder are committed to short-term memory. The first of the three chunks, 'A', is expanded to produce three further codes, 'S', 'B' and 'J'. As before, the first of these codes is expanded whilst the second and third are stored in short-term memory. However it is evident that this last level of codes is nonreducible, and can now be decoded to the letters from the sequence that they represent. Having first extracted the letter 'S' from its coded representation, the subject recovers the most recent item committed to short-term

memory, the code 'B'. Once again, this is decoded to extract the letter that it represents and the next most recent code retrieved from short-term memory (figure 3.6).

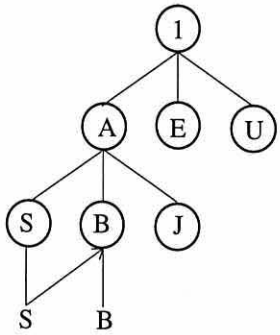

Figure 3.6 *Decoding operation for recalling a nine letter sequence (SBJ...) organised in three chunks (AEU) (Adapted from Johnson, 1970)*

However, having expanded the first chunk, 'A', in its entirety, the subject must once again retrieve the most recent item committed to short-term memory, the code, 'E'. This process is continued until the hierarchy has been expanded and the nine items recovered from memory in sequence.

Johnson suggests that this decoding model can be formalised as a set of six postulates:

(1) It is possible for a code to be recalled provided that it represents at least one item of information.

(2) Whenever a subject recalls a code, he decodes it into the components at the next lower level.

(3) After a subject has decoded a code into its components, he holds the components in a temporary memory store, except for the component whose ultimate members have temporal priority in the sequence. That component is further decoded.

(4) Whenever an overt response is produced, the subject returns to his temporary store and recovers the code whose ultimate members have temporal priority over the other codes. That code is then decoded.

(5) A code is not analysed for the information it contains until it is decoded.

(6) Whenever a subject is uncertain regarding a decoding step, he completely terminates his response attempt.

Johnson explains that a code, containing both item and order information for a number of items, could be considered as an opaque container - opaque in that recovery of the code does not allow the subject to recover the items within that code (container) immediately without further decoding. The notion that the code can be considered as a container holding item information is similar to that outlined for previous non-associative models (e.g. Conrad, 1965) - except for the obvious difference that item *and* order information are held in Johnson's codes, while Conrad's bins contain only item information, the order being inferred from the relative serial position of the bin containing the item. Also, if the code is lost from memory, all the information "hidden" by that code is deemed irretrievable to the subject (Johnson, 1969)

Although it has already been stated that Johnson's model stores both item and order information in each chunk, it is as yet unclear as to how the order information is represented. A number of possibilities are suggested:

(1) Order could be inherent in the organisation itself, i.e. chunking could imply ordered recall.

(2) Subjects could form weak inter-item or inter-code associations during learning. However, this would have to be refined as only weak associations would exist between the last item of one chunk and the first item of the next.

(3) Codes could be tagged with order information when the subject stores them in memory. A subject's ability to retrieve the position of an item would not depend on the subject's ability to recall any previous item with any degree of accuracy.

This third hypothesis appears to be very similar to that suggested by Conrad (1965) and is the method by which Johnson models serial-order effects.

Johnson's model relies on a measure of performance, the transitional-error probability (TEP) which is defined as the probability that an item in position i+1 will be recalled correctly given correct recall of item i. The TEP can go across chunk boundaries. Johnson reports that TEPs at chunk boundaries are consistently higher than they are within chunks which suggests that the problem during recall is reinstating a chunk rather than the items contained within (Murdock, 1974).

In summary, Johnson (1970) outlines a model of short-term memory based upon the concept of chunking (Miller, 1957). Johnson describes a hierarchical structure of opaque containers, or codes, that contain both item and order information. Recall is all or nothing, if a code in the hierarchy is forgotten, all the paths below that code and their contents, are forgotten. However, it is unclear how this model could be implemented as the encoding and decoding processes are largely unspecified, particularly with distributed item representations and the associative mechanisms outlined in chapter 1.

3.4.2 Perturbation model

Estes (1972; Lee, 1992) considers the forgetting of item and order information and the effects of phonemic similarity, reviewing that when item errors occur (i.e. no loss of order information), then the letter replacing the correct letter is typically drawn from a similar acoustic confusion set (Conrad, 1964). Furthermore, order (transposition) errors are produced in greater proportions than item errors, the former being distributed about the central serial positions, the latter, uniformly across each serial

position (Bjork & Healy, 1970). Having rejected a number of theories of order effects: coding (e.g. Johnson, 1970), item-to-item association (e.g. Wickelgren, 1965a), positional-coding (e.g. Conrad, 1965; Johnson, 1970); Estes outlines a new model for association.

Estes suggests that there exists in memory a pool of elements he calls *control elements*. If two items are presented in succession to the subject, then in contrast to an inter-item association model where the two items would be associated together in sequence, Estes suggests that the two items are instead associated to a control element (e.g. figure 3.7).

Figure 3.7 *Neighbouring items connected to a control element (Adapted from Estes, 1972)*

A new control element is established at each discontinuity in the input sequence: at the beginning of a new chunk, for example, or between words in a sequence. Therefore, it is possible to develop a hierarchical structure constructed using associations between control elements or features. For example, consider learning a sequence containing a pair of letters with two features in each letter. The letter-features would be associated with a letter control element. Both letter control elements would then be associated with one overall sequence control element. Estes suggests that the control element has some temporal context component which must evolve over time until it may be stabilised through association with some element in long-term memory.

Thus far, the model describes the storage of item information in a hierarchical structure not dissimilar to that outlined by Johnson (1970). However, there is no reference to the storage and retrieval of serial order information. Estes proposes that the problem of serial order can be divided into two sections: the first, the decay over time of a short-term representation of order; and the second, the establishing of a stable representation of order in long-term memory by rehearsal. The model addresses

the first of these by the inclusion of a reverbatory loop between the item representation and the contextual control element:

Figure 3.8 *Reverbatory loop for decay (Adapted from Estes, 1972)*

Here, the reverbatory loop reactivates the item at a rate determined by a phase within the system. If the control element is associated with more than one item then a reverbatory loop exists for each of the items, hence the items can be reinstated in sequence. It is because the phases of these reverbatory loops will alter over time due to random error and interference from other neuronal processes, that the timing of individual items will deviate sufficiently to introduce order errors into the system. If items are presented during the period between the list being presented and recall, perturbations in the reactivation cycles of the reverbatory loops may introduce order, intrusion and omission errors. Estes assumes that loss of order information is primary, and at a greater rate for closely spaced and similar items, with loss of item information derived from the loss of order.

Estes applies the model to replicating the serial position and distance function curves of Healy (1971, cited in Estes, 1972) which it manages to predict reasonably well. Estes also presents an account of chunking (Miller, 1956; Johnson, 1970) and in particular how a chunk size of three or four items is optimal (Wickelgren,1964, 1967).

In summary, Estes (1972) presents a model of serial order in which item information is stored in a hierarchy of item to control element associations (cf. Johnson, 1970). Order information is stored in the form of reverbatory decay loops between each item and the control element with which it is associated. Estes demonstrates that a model of this form is capable of reproducing the item and order serial position curves and distance functions of Healy (1971, cited in Estes, 1972) and Bjork and Healy (1970). It also provides an account for why chunk (Miller, 1956) sizes of three or four items produce

optimal results (Wickelgren, 1964, 1967). However, like the chunking model of the previous section, it is unclear how this could be model data at the level of individual trials with distributed representations.

3.5 Dynamic context models

3.5.1 Competitive Queuing model

The competitive queuing (CQ) model is a connectionist model for sequence learning and recall that has been applied successfully to the problem of learning and recall of English monosyllabic words (Houghton, 1990, 1994a, 1994b; Houghton, Glasspool & Shallice, 1994).

There are five main constraints placed upon the CQ model when applied to phonological retrieval of word forms:

(1) Recall of a word should involve the network activating a sequence of states so that each phoneme of the word becomes the most active. In this manner, words are not stored as a copy or template, but recreated on-line as a dynamic activity pattern.

(2) There should be no position specific coding of elements as these lead to problems when attempting to represent temporal order.

(3) Phonemes should be pre-activated before being produced, and the ability to do so should be a prerequisite of the model.

(4) The network should learn sequences through exposure to them followed by associative weight changes. During recall, excitatory feed-forward connections ensure that only learning of positive weights takes place.

(5) Inhibitory mechanisms should be involved in all levels of operation.

The CQ model is a three layer architecture of *on-centre* (i.e. feedback positively to themselves) *off-surround* (inhibit and are inhibited by other units in the same layer) nodes (Houghton, 1990, figure 11.2). The model uses local representations to represent, for example, linguistic items such as phonemes. Node activations vary between 1 and -1, with 0 representing the background activation and negative values representing suppressed activation.

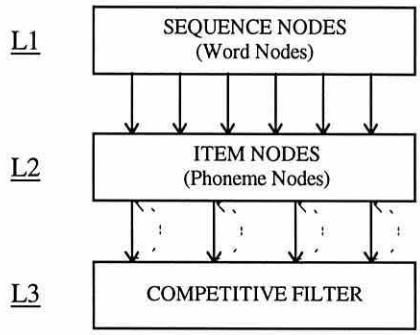

Figure 3.9 *Inter-layer connections in basic CQ model*
(Adapted from Houghton, 1990)

Figure 3.9 illustrates the basic CQ model. Each node in the first layer (L1), the *word node* layer, represents a learned sequence of phonemes in L2. The first and second layers are fully interconnected with excitatory connections in both the feed forward and feed back directions. The weights on these connections are variable as it is here that learning takes place when a new word is presented.

The first layer also contains pairs of initiator and end nodes (I-nodes and E-nodes accordingly). The connections between L2 and L3 are feed forward one-to-one excitatory and essentially map the second layer on to the third layer. However, the lateral interactions between nodes in L3 are stronger than those in L2 making the layer a "winner-takes-all" architecture. It is possible that a number of nodes may be active in the second layer at any one time. Once an item has *won* in the third layer, a strongly inhibitive feedback connection ensures that that same item is inhibited in the second layer. In this manner, the third layer acts as a competitive filter.

Learning is a two stage procedure. The first is the initial exposure to the word and the phonemes required in order to reconstruct that word. Initially all the weights between the first two layers are set to zero (the background value). A word is represented as a sequence of delta vectors (zero in all but one of its elements, which takes a value of +1) that correspond to phoneme nodes in the second layer. At the onset of each new word, an I-node is fully activated and then decreases with each successive time step. The activation of each phoneme node in the second layer, a_i, is computed as follows:

$$\mathbf{a}_i(t) = \min(1, \delta \mathbf{a}_i(t-1) + \mathbf{I}_i(t)) \tag{3.3}$$

Here I_i represents the value of the ith element of the current input vector and δ, the decay factor. Hence the activation of each node in the second layer either stays at zero or, once activated (to +1) by the corresponding unit in the first layer, decays with time to zero. At the end of the sequence, the E-node becomes fully activated while the I-node has decreased to approximately zero activation. This is achieved by the inclusion of rate of change of activity detectors in the input pathway that detect when a new word is presented (and hence activate an I-node) and also when it has finished (and activate the corresponding E-node).

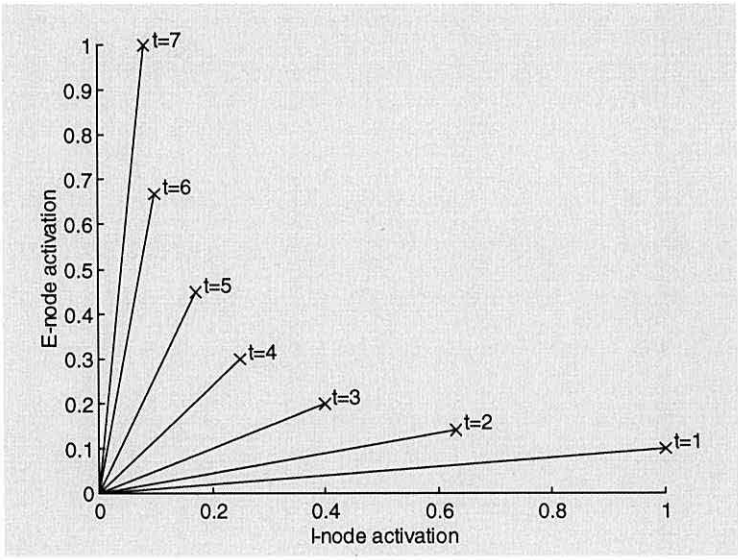

Figure 3.10 *I-node and E-node activations for a seven item sequence (Adapted from Houghton, 1994b)*

The combination of I and E-node signals allow serial position to be represented by a continuous distributed state (figure 3.10) which can provide temporal edges with which to bound the start and end of each word. Although only two dimensional, the signal could be of any dimensionality, however Houghton, Glasspool and Shallice (1994) argue that a larger dimensionality control signal would be harder to account for.

Learning in the weighted connections between the first two layers is by simple Hebbian association where a_i represents the activation of the ith phoneme node and $\mathbf{a}_j$ represents the activation of the jth I-node in the word layer:

$$\Delta \mathbf{w}_{ji}(t) = \lambda \mathbf{a}_i(t)\mathbf{a}_j(t) \tag{3.4}$$

As the word ends and the E-node activates, so the layer two activations are all decayed by a single time step and the weights from the E-node to the units in the second layer are changed according to the previous equation. This initial period of learning is unsupervised and is sufficient for shorter stimuli (Houghton, Glasspool & Shallice, 1994). However, learning is a two stage process and the second, practice, stage involves supervised learning.

During practice, the weights that were formed in the initial exposure stage are altered by a process of supervised learning, until the model can reproduce the phonemes in the correct order. During this phase, the I-node corresponding to the target word is activated and the output from the competitive filter compared to the target word after each time step. If there is any error at the output, the top-down weights from the start and end nodes are changed. Weights to the appropriate response are increased while those to the incorrect response are decreased (Houghton, 1990, p. 300). Note that no error term is propagated during this process. As the I-node is most strongly associated with elements at the start of a sequence, and initial access to a sequence is by activation of the I-node, so the beginning of a list is the most readily recollected portion of a sequence.

Having learned a number of words, the system can be made to recall the items by setting the activation of the appropriate I-node to one. The activation feeds through to the second layer where phoneme nodes are updated as follows:

$$\mathbf{a}_i(t+1) = \begin{cases} \delta \mathbf{a}\ (t) + (1 - \mathbf{a}_i(t)) f(net_i(t)), & \text{if net}_i(t) > 0 \\ \delta \mathbf{a}\ (t) + (1 + \mathbf{a}_i(t)) f(net_i(t)), & \text{if net}_i(t) < 0 \end{cases} \tag{3.5}$$

Here, δ represents a passive decay parameter, $(1 \pm \mathbf{a}_i(t))$ corresponds to a gain control which ensures that the increase in activation is proportional to the current activation, the function f represents a squashing function, such as the sigmoid function, that ensures that the function net_i remains bounded between 1 and -1. Of great significance is the relationship between the I-node and the E-node. It has already been stated that the I-node is most active at the start of the word, and least at the completion of that word, the converse for the E-node. This is most simply achieved by the relationship:

$$\mathbf{a}_{E-node}(t) = 1 - \mathbf{a}_{I-node}(t) \tag{3.6}$$

Furthermore, the issue of how a word is terminated is addressed by the inclusion of a inhibitory control circuit that activates once activity in the phoneme level drops below a threshold value. Once active, it resets the word node activations to zero to prevent further phoneme nodes becoming activated. Hence, during recall, after an I-node has been activated and the activation propagated forward through the phoneme layer, the first items will have the greatest levels of activation. As the activation passes to the competitive filter layer, the most active unit will suppress the remainder before feeding back to the corresponding unit in the phoneme layer and deactivating itself. Meanwhile, the level of activation of the winning unit in the competitive filter layer will begin to decay rapidly.

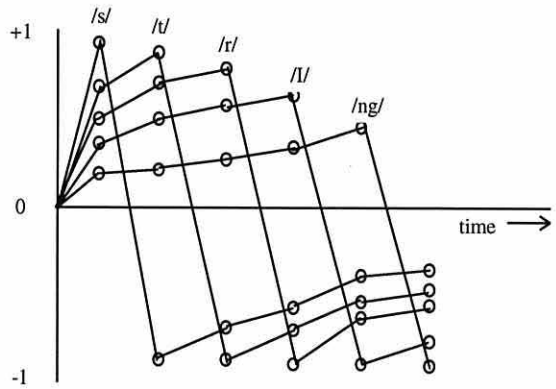

Figure 3.11 *The activation history of phoneme units during the recall of the word /strIng/ (Adapted from Houghton, 1990)*

Figure 3.11 illustrates a sequence of phonemes becoming the most active in the competitive filter layer during recall of the word *string*. Each curve is labelled by the phoneme it represents at the point it wins the competition in the competitive filter and is subsequently suppressed. It is clear from this figure also Houghton (1990, figures 11.8, 11.9 & 11.10) that:

(1) All the sequence elements, or phonemes, become activated in parallel.

(2) Elements become gradually more activated with time until recall at which point they are rapidly suppressed. The gradual increase in activation is due to both the decrease in lateral inhibition and the increase in the activation of the sequence's E-node.

(3) The relative activations at any particular time step should reflect the order in which the elements will be recalled.

Recall stops when either the correct number of items has been recalled, or when the average activation of all the nodes falls below a certain threshold, or finally, when a special "end of sequence" node becomes active.

Subsequent refinements include a *geminate* node that facilitates the modelling of repeated items in adjacent serial positions (Houghton, Glasspool & Shallice, 1994, p.

385). These have to be identified and tagged accordingly during a pre-processing stage prior to learning.

Houghton, Glasspool and Shallice (1994) attempt to fit the CQ model to word length data. However performance is poorer than the empirical data for each word length condition. Houghton attributes this to either the accumulation of noise which will decrease performance on later list items, or the limitations of a two dimensional control signal, particularly for longer items where seven separable events must be represented within the 90 degree phase space (figure 3.10). In fact, this last statement would suggest that the memory span of the CQ model occurs as a direct result of the limited discriminatory power of a two dimensional control signal.

Houghton, Glasspool and Shallice (1994) also observe that the CQ model can produce a similar range of errors (insertions, deletions, exchanges, shifts and substitutions) to that reported in the empirical data. The CQ model also produces reasonable serial position curves, although Houghton observes that the peak of the error distribution curve occurs nearer to the end of the sequence than is apparent in the empirical data.

In summary, Houghton (1990, 1994a) outlines an architecture for serial ordered recall that relies on a competitive queueing mechanism. The model uses a two dimensional control signal, implemented by I and E-nodes whose activations decay and rise in a manner which facilitates the modelling of the temporal order. A competitive filter selects the most active output node and inhibits all other output nodes during recall, before deactivating the representation of itself in the second layer and as such removing itself from the queue of potentially recallable items. The model is shown to produce realistic word length, serial position curve and error distribution data (Houghton, Glasspool & Shallice, 1994). However, the author would suggest that the limited ability of a two dimensional control vector to produce highly discriminable control states is responsible for the memory span performance. Furthermore, the use of such a limited capacity control signal facilitates the need for a more comprehensive inhibitory mechanism at recall.

3.5.2 Network model of the Articulatory Loop

In the following section, a modular model of working memory, the articulatory loop (Baddeley & Hitch, 1974; Baddeley, 1986) is outlined, then the specific network implementation of Burgess and Hitch (1992) is described.

Baddeley and Hitch (1974) and Baddeley (1986) outline a modular model of working memory that contains a number of different components including the central executive with its peripheral components: the visuo-spatial scratch-pad and an articulatory loop for processing language material. However, the main focus of attention for this section is the articulatory loop for phonological information. In its most basic form, Baddeley's articulatory loop can be described as a limited capacity phonological store, analogous to a short loop of tape, coupled with a sub-vocal rehearsal process. It is assumed that memory traces will decay, unless rehearsed, after a period of one or two seconds.

There is much evidence to support the notion of a phonological store in memory[7]. First, substitution errors that occur in a visually presented immediate recall task are similar to those that occur when presentation is auditory (Conrad, 1964). This led to the discovery of the phonemic similarity effect: that phonemically similar items are harder to recall in order than phonemically nonconfusable items (Conrad & Hull, 1964; Wickelgren, 1965a; Baddeley, 1966a).

Evidence is also provided by the results of experiments that used articulatory suppression. Briefly, subjects are presented with sequences of acoustically confusable and nonconfusable items. Rehearsal through articulation is suppressed by requiring the subjects to vocalise a distractor (e.g. the word *the*; Murray, 1965, 1967). Only if the stimuli are presented visually, and rehearsal prevented in this manner, does the phonological similarity effect vanish (Murray, 1968). Further evidence for a phonological store is provided by word length effects (Baddeley, Thomson & Buchanan, 1975) which demonstrate that subjects find that they can recall more

[7] See chapter 2 for more details of the empirical data described briefly here.

shorter words than longer words from short term memory. A similar effect was reported by Ellis and Hennelley (1982) who demonstrated that bilingual subjects had a poorer memory span for Welsh digits than for English digits, which they subsequently attributed to the lengthier articulation times for Welsh digits. Suppression removes the word length effect for both visually presented and spoken stimuli (Baddeley, Lewis, & Vallar, 1984). This suggests that the articulatory loop must be refined in order to account for modality effects and phonemic similarity and word length.

In this basic form, the articulatory loop has been shown to account for a range of findings (see Baddeley, 1986; Burgess & Hitch, 1992). However, it is unclear precisely how the model can be applied to the problem of serial order. Burgess and Hitch (1992) note that the phonemic similarity effect is explained by the articulatory loop in terms of the difficulty in being able to discriminate between the memory traces of similar items, much in accordance with Wickelgren (1965a). However, they observe that there is little explanation for why similar items transpose (Conrad, 1964) nor how a bowed serial position curve may result during ordered recall. Further, given the existence of data such as serial order intrusions (Conrad, 1960a) which may be accounted for by a positional model of order memory, Burgess and Hitch suggest a modification to the articulatory loop which includes both item-to-item and position-to-item associative mechanisms in order to account for serial order data.

Therefore, Burgess and Hitch (1992) describe a network implementation of the articulatory loop, aiming to account for a series of empirical findings including the reverse-S shaped memory span curve, phonemic similarity, word length and articulatory suppression effects. They also wish to replicate the basic bowed shape of the serial position curve, order error distributions and phonemic confusion errors.

The network, outlined in figure 3.12, is a four-layer feed-forward network with a feedback *articulatory* loop between the output phoneme layer and the input phoneme layer. Items are activated by phonemic input during presentation and reactivated by context and phonemic feedback during output.

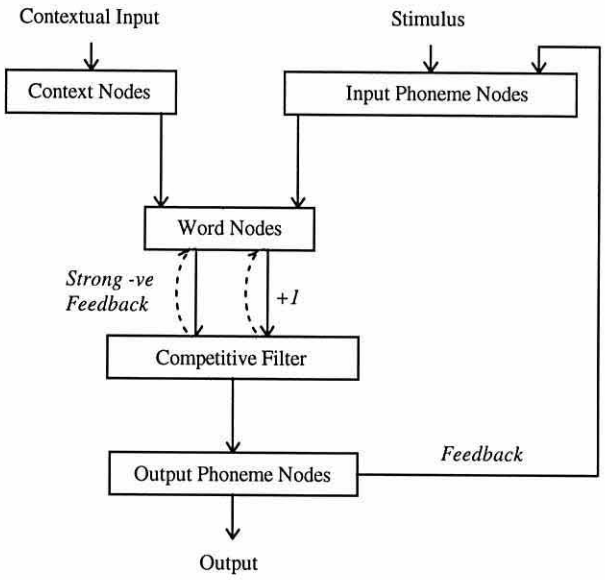

Figure 3.12 *Outline of the network model of the Articulatory Loop (Adapted from Burgess & Hitch, 1992)*

The contextual input for an item is represented by a pattern of activation that evolves with time. The context represents nonphonological information, including temporal information. As each word is presented, two-thirds of the context vector elements, selected at random, are updated. Of those that are updated, a random subset of six are assigned non-zero activations, while the remainder are zeroed. In this manner, the similarity between neighbouring contexts varies with their temporal separation.

Items are represented at the input and output locally by phoneme nodes. There are 53 input and output phoneme nodes, one for each phoneme. The activation of each (non-context) node depends upon the activations of the nodes which are connected to it and the weights connecting them to it. Activations take continuous values between -1 and +1.

Selecting a word during recall involves a cyclic competitive queuing process (Houghton, 1990). Each word node is connected to a competitive filter node by an excitatory connection. However, there is strong lateral inhibition between the nodes in the competitive filter which results in the node from the competitive filter that is connected to the most active word node, suppressing all of the remaining competitive

filter nodes. The winning node then excites the corresponding output phoneme nodes. Using a set of strongly inhibitive feedback connections from the competitive filter to the word layer, the current word node may be suppressed when the word layer next becomes active by the winning competitive filter node.

The weights responsible for the lateral inhibition in the competitive filter, also the one-to-one excitatory connections from word nodes to competitive filter nodes and the one-to-one inhibitory feedback from the competitive filter to the word nodes, are all fixed. Learning only occurs in the temporary excitatory weights between the context and word nodes, and also between the output phoneme and input phoneme layers in the feedback, chaining, loop. Learning is by "one-shot" Hebbian adjustment and both sets of weights decay after each phonemic time step. Noise is added to the learned weights in order to introduce errors. Relearning of the temporary weights during recall ensures that the weights do not decay to zero.

The word layer contains 26 nodes corresponding to each letter of the alphabet. The weights connecting the relevant phonemes from the input phoneme layer and the word node are prelearned. In this manner, in order to learn to recognise the letter 'c', the input phoneme nodes for "s" and "ee" and the word node "c" must be activated and the weights updated accordingly. However, other word nodes that share either phoneme in common will also be activated (e.g. "b", "g", "s" or "x"). Phoneme nodes are not ordered and therefore the model treats "s-ee" and "ee-s" as equivalent. If words in a list share phonemes, erroneous activation of input phoneme nodes may occur.

However, one of the most significant elements of the model is the ability to vary the degree to which it is either a chaining model or a context driven model. F_{ph} determines the fraction of the input which comes from the phonological store rather than the context layer. When the input phoneme layer is driving the word layer, F_{ph} tends towards unity, however when it is driven by the context layer, F_{ph} tends towards zero.

The model, as described here, is applied to a number of empirical paradigms. The first considers the models capacity measured in terms of memory span. Levels of

performance are comparable to those exhibited by humans and are determined by the level of noise in the system. Results confirm that the model (F_{ph}=0.5) has a suitable memory span and exhibits a phonemic similarity effect (Burgess & Hitch, 1992, figure 5). Further, when errors occur, they occur between phonemically similar items (Conrad, 1964). A word length effect, where time to articulate a word is proportional to the number of phonemes, is also reported.

Next the model (F_{ph}=0.5) is applied to the problem of reproducing the serial position curve. However, results report a significant lack of any recency effect. However, once again there is a phonemic similarity effect when items are selected from a vocabulary of similar letters. However, the lack of recency is still apparent in these curves.

Analysis of the different categories of error produced by the model reveals that order errors occur in much greater proportions than item errors (Bjork & Healy, 1972), particularly when the model is configured as a chaining-only model (F_{ph}=0.98). However, the model is incapable of producing omission errors without a modification to the architecture. When this modification is made, omissions occur towards the end of the list. The order error distribution about the target serial position is shown to depend upon the extent to which the model is a chaining based account (Burgess & Hitch, 1992, figure 11). When this is the case (F_{ph}=0.98), transpositions are not clustered about the target serial position, however, when the context layer alone drives the model, transpositions occur more naturally (cf. Estes, 1972; Bjork & Healy, 1972; Henson, Norris, Page & Baddeley, 1996). In summary, the model produces a realistic distribution of errors, more order errors than item errors and phonemic similarity order errors. Transpositions occur naturally only when the degree of chaining in the model is minimal, although even in this case, the separation between transpositions appears unnaturally large.

When Burgess and Hitch attempt to fit the network model of the articulatory loop to the phonemic similarity effect of Baddeley (1968, experiment 5), it fails to reproduce the "sawtooth" serial position curves required of the data (Burgess & Hitch, 1992, figure 15). Burgess and Hitch account for this poor performance by explaining that

when phonemically similar stimuli are used, there is as much chance of an error between the two items following the similar items as it happening between the items themselves. This is due to the chaining system within the phoneme layers.

Burgess and Hitch suggest a number of modifications to the model's architecture in order to improve performance including a modification to their context signal. In order to reduce the number of transpositions occurring between widely temporally separated list items, it is suggested that a context is employed that has a zero correlation over these larger distances. Introducing a set of contexts which overlap between each temporal step resulting in a gradually decreasing similarity between successive states, allows a recency component to return to the serial position curve (F_{ph}=0.02). This is due entirely to the increased difference between the last context and the earlier contexts. This also reduces the number of widely separated order errors (Burgess & Hitch, 1992, figure 17).

A later implementation of the articulatory loop, a three layer connectionist architecture (Burgess, 1995; Burgess & Hitch, 1996), is fitted successfully to a range of empirical data including serial position curve, memory span and the phonemic similarity effect.

In summary, the Burgess and Hitch (1992) implementation of the Baddeley (1986) articulatory loop model further illustrates the limitations of a chaining based account of serial order (cf. Lewandowsky & Murdock, 1989). They observe that the model performs best when there is little or no chaining relying instead on item-to-context association (Burgess, 1995; Burgess & Hitch, 1996).

3.6 Activation gradient models

Highlighting the inability of chaining models such as TODAM (Lewandowsky & Murdock, 1989) and the network model of the articulatory loop (Burgess & Hitch, 1992) to address fundamental data such as the sawtooth serial position curves of acoustically confusable and nonconfusable items (Baddeley, Papagno & Norris, 1991;

Burgess & Hitch, 1992), Page and Norris (1995; Henson, Norris, Page & Baddeley, 1996) outline a non-chaining model of immediate memory, the Primacy model.

The primacy model stores order information in terms of the relative activation levels of list items. Items are represented locally by nodes and the temporal order of each item is represented by the pattern of activation across each node. Effectively, each item is associated to the start of list: the first item most strongly, then less so for each of the remaining items (figure 3.13).

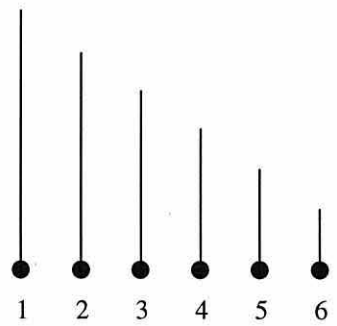

Figure 3.13 *Primacy gradient for a six item list (Adapted from Page & Norris, 1995)*

More specifically, the activations across a four item list, *ABCD*, may be defined as:

$$x_A > x_B > x_C > x_D > 0, \tag{3.7}$$

and

$$x_A - x_B = x_B - x_C = x_C - x_D \tag{3.8}$$

Where x_i represents the activation of the node representing item **i**. Thus the ordering of the node activations corresponds to the temporal ordering of the stimuli and is termed the primacy gradient. This means the during recall, the primacy gradient need only be established the once by cueing with a "pre-list" context (Henson, Norris, Page & Baddeley, 1996) in order to activate each item node in parallel.

Once established, ordered recall of each item is possible without the need for a separate recall cue for each position (cf. Burgess & Hitch, 1992). Recall is simply a iterative process of selecting and then suppressing the item corresponding to that with the greatest activation. Suppression could be achieved in a number of ways including the competitive filter of Houghton (1990). Page and Norris suggest that without the addition of gaussian noise, recall may be perfect.

At present, there is only one free parameter in the primacy model: N which corresponds to the ratio between the inter-activation step (x_A - x_B) and the amount of noise added to each activation prior to selection. The model (N=2) can be shown to possess single item primacy and recency effects and a very high mean performance for a six item list (Page & Norris, 1995, figure 2). Furthermore, in accordance with the empirical data, the majority of the errors that occur are paired transpositions concentrated in the central list positions. However, it is with the addition of a second parameter introducing decay to the primacy gradient, D, that the model can be fitted to experimental data such as the lists of phonemically similar and dissimilar items of Baddeley (1968, experiment 5) and Henson, Norris, Page and Baddeley (1996, experiment 1) phonemic similarity effect data (Page & Norris, 1995, figures 14 & 15).

Page and Norris observe that, unlike TODAM (Lewandowsky & Murdock, 1989) or the Burgess and Hitch (1992) model, the primacy model produces a desirable type of error referred to as *fill-in*. Fill-in errors occur when an early list item (e.g. *B*) is recalled in the serial position that corresponds to item *A*. Subsequently item *A* will most probably be recalled in the correct position for item *B*, ensuring that transpositions occur maximally for items separated by one serial position (e.g. Estes, 1972). This is in contrast to Burgess and Hitch's (1992) model where if item *B* was again recalled incorrectly (and then suppressed) in the first serial position, the context cue for the second serial position would activate both items *A* and *C* with equal strength. If *C* is in fact recalled, so the probability of recalling *A* will then decrease as each cue becomes increasingly dissimilar to the cue for *A*. This means that *A* may not be recalled until the last serial position where it will be the only response left

unsuppressed. Clearly this will reduce the degree of recency that such a model could exhibit (Henson, Norris, Page & Baddeley, 1996).

It is evident therefore, that primacy occurs due to the nature of the primacy gradient and that as recall progresses and the primacy gradient decays, so more errors will be reported in the later list positions (Henson, Norris, Page & Baddeley, 1996). Introducing item errors, by use of a noise threshold above which items are recalled and below which they are omitted, Page and Norris can reduce the amount of last item recency to acceptable levels. This requires two further parameters: P, the peak value of the primacy gradient at the start of recall; and T, the threshold noise. With the addition of these two parameters, Page and Norris anticipate that the probability of an item error increases in each list position. The four parameter model can provide a very good fit to both the Baddeley (1968) and Henson, Norris, Page and Baddeley (1996) data (Page & Norris, 1995, figure 6). However, the distance functions produced in this same simulation (Page & Norris, 1995, figure 7), show few transpositions occurring between items in adjacent serial positions - the curves are too pointed. Page and Norris fit the primacy model to memory span data demonstrating both the characteristic reverse 'S' shaped curves with realistic values for memory span and also linear rate of articulation effects (Page & Norris, 1995, figures 8 & 9).

However, most significantly, the model is fitted to the phonological similarity effect data of Baddeley (1968) and Henson, Norris, Page and Baddeley (1996). In order that a phonemic confusion effect be produced, the primacy model requires the addition of a second stage of processes before output. As before, an item is selected from the first stage, however now this activates a series of nodes corresponding to local representations of all possible outputs. In this manner, phonologically similar items will all be activated by some amount S when a similar item is forwarded to this layer. Dissimilar items will be unaffected by the layer. Before selection can be made, the output activations are multiplied by the corresponding primacy gradient activations in order to ensure that confusions also occur in accordance with the transposition locality gradient observed empirically (Henson, Norris, Page & Baddeley, 1996). Furthermore,

response suppression must occur independently at both stages in order to maintain the fill-in effects described previously.

In summary, Page and Norris (1995) outline a model of immediate memory, the primacy model. Items are represented locally by nodes, the degree of activation of which determines the order in which they are recalled. This, primacy gradient of activation, decays over time in order to increase the likelihood of confusion in the later list positions. Recall involves the selection of the item with the greatest level of activation followed by immediate suppression to prevent it being recalled repeatedly. A second layer of nodes is introduced in order to facilitate the modelling of phonemic confusion effects such as those presented by Baddeley (1968, Experiment 5). The model is shown to produce realistic serial order performance with a minimum of parameters, although additional gaussian noise on the activations is required in order to introduce item errors during recall. The model is fitted to a number of empirical results including memory span, articulation rates and the phonemic similarity effect (Henson, Norris, Page & Baddeley, 1996). However, it should be observed that although the primacy model provides a good account of the data, it does so by making the assumption that there is prior knowledge of the list length. Examination of equation 3.8 suggests that in order to ensure that the difference in the primacy gradient of activation between adjacent items is constant for the whole list length, knowledge of the number of items in the list is required to ensure that the activation of the later items does not fall to zero. This may be clarified by considering figure 3.13 and extrapolating the gradient for a further item: the activation would be approximately zero.

3.7 Summary

In this chapter, a number of different models and accounts of short-term memory for serial order have been presented and considered in light of the empirical data that they can account for, and that which they fail to address.

Bin models

In section 3.2 we summarised the item-to-position "bin" theory proposed by Conrad (1965) as an attempt to address both serial order intrusion errors (Conrad, 1960a), vocabulary size effects (Conrad & Hull, 1964) and phonemic effects during erroneous recall (Conrad, 1964). We observed that Conrad's hypothesis that order errors are in fact pairs of item errors could not account for Healy's (1974) finding that order and item errors occur in different distributions across serial positions. Conrad's account also fails to explain why transposition errors occur in such large quantities in short-term memory (e.g. Wickelgren, 1965a) and why loss of order information is faster than loss of item information (Bjork & Healy, 1970). We would also suggest that it is unclear why each "bin" should be searched in the correct serial order during recall.

Chaining based models

In section 3.3 we considered formal models that employ item-to-item associative chaining (e.g. Ebbinghaus, 1913; Lewandowsky & Murdock, 1989). Recall relies upon at least one of the previous items being used as the cue for the next (Wickelgren, 1965a; Jordan, 1986). Wickelgren (1966) suggested that such a model could account for order errors occurring after repeated items in sequences (Wickelgren, 1965c). However, chaining based models are unable to predict the phonemic similarity effect of Baddeley (1968, Experiment 5) as they predict that items following confusable items suffer during recall.

TODAM, a distributed chaining-based model of serial order that uses convolution and correlation (cf. Metcalfe Eich, 1982) to perform serial order learning, is shown to fit a number of serial order paradigms including memory span and the serial position curve (Lewandowsky & Murdock, 1989). However, attempts to fit TODAM to Baddeley's (1968, experiment 5) phonemic similarity data by Baddeley, Papagno and Norris (1991) reaffirm the observation that chaining models are not suitable for addressing such data (Henson, Norris, Page & Baddeley, 1996). Further inadequacies of TODAM include skewed transposition gradients and the sampling without replacement scheme (Nairne & Neath, 1994). Also, there is the arbitrary relationship between the item and order weighting parameters and item list position, and the manipulation of the number

of recallable items in order to introduce recency to the serial position curve (Mewhort, Popham & James, 1994), which clearly contradicts the empirical findings (Drewnowski, 1980).

Recurrent networks (Jordan, 1986; Elman, 1990) are also chaining based accounts and are similarly flawed as a result. However, they learn sequences by repeated presentation and back-propagation and are therefore not suited to single-trial learning tasks.

Hierarchical models

A third category of models of serial order employ recoding of the stimuli into a hierarchical structure of memorial item and order information (Miller, 1956; Johnson, 1970). An extension of this type of model is described by Estes (1972) in which item information is stored in a similar hierarchical form. Order information and forgetting is dependant upon a reverbatory loop between each item and the context associated with it. Estes demonstrates that such a model can provide a good account of chunking, in particular the finding that the optimal size for a chunk is three or four items (Wickelgren, 1964, 1967). Most significantly, Estes also demonstrates that the model can reproduce the distance functions typical of the empirical data (Healy, 1971; Bjork & Healy, 1970). Although these models provide a good theoretical account, it is unclear how they could be implemented to perform single trial trial-level learning and recall using distributed representations.

Dynamic context models

Section 3.5 reviews current models of serial order that employ the use of item-to-context association in order to model sequential behaviour. Houghton's competitive queuing network (1990, 1994a, 1994b; Houghton, Glasspool & Shallice, 1994) employs the use of a two dimensional control signal. The three layer architecture uses local representation at the phonemic level, each phoneme being associated with a pair of sequence nodes that represent the relative position of the current node in relation to start and end of the word. The network uses a competitive filter to select and then inhibit the most active item node. Houghton, Glasspool and Shallice (1994)

demonstrate that the model is capable of producing a reasonable fit to empirical data including word length effects, serial position curves, error distributions, and, with a modification to the network, lists containing repeated items. However, it would appear that the memory span of the network is determined in the main by the limited discriminatory capacity of the two-dimensional context signal. This limited dimensionality control signal also facilitates the need for a powerful inhibitory mechanism such as the competitive filter.

In their network model of the articulatory loop, Burgess and Hitch (1992, 1996; Burgess, 1995) describe a multi-layer architecture that uses local representations of phonemes and a context signal that evolves with time. Selection is implemented with a competitive filter (Houghton, 1990) that selects the most active output node, inhibiting rival nodes, before inhibiting itself in order to prevent repeat errors in successive serial positions. Furthermore, the Burgess and Hitch model employs both item-to-item chaining and item-to-context associations, the degree of each being controlled by one of the model's parameters. The authors observe that the model performs best when there is little or no chaining. Burgess and Hitch fit the network model of the articulatory loop to a range of empirical data including the serial position curve, item and order errors and distance functions. However, like TODAM, the original model fails to reproduce the "sawtooth" serial position curves of Baddeley (1968, experiment 5) although a refined version proves successful (Burgess, 1995; Burgess & Hitch, 1996)..

Activation gradient models

Finally, we present the primacy model of Page and Norris (1995). This is a simple model that relies on a gradient of activation produced by cueing for recall only once, that activates items in decreasing amounts as displacement from the first serial position increases. The most active node is recalled first then inhibited, and then the next most active is recalled, and so forth for the complete sequence. The model in its simplest form is shown to reproduce the serial position curve. However, with a noise component, added to introduce item errors, the model is fitted to memory span and articulation rate data. With the addition of a second stage of processing, the model is

shown to reproduce the serial position curves of Baddeley's (1968, Experiment 5) phonemic similarity data (Henson, Norris, Page & Baddeley, 1996). However, the manner in which the primacy gradient is generated would suggest that the model requires some apriori knowledge of the number of items in the sequence in order to ensure that the primacy gradient does not fall to zero before the end of the list. Also, analysis of the transposition gradients reveals that the primacy model does not produce a high proportion of errors with items from serial position immediately adjacent to the target serial position.

Conclusion

It is clear that none of the formal models described in this chapter can account for all of the adult data. Also, it is unclear how any of them could provide a developmental account of short-term memory. Therefore, we will now present two novel architectures for association and serial order recall.

CHAPTER 4

Developmental Associative Recall Network (DARNET)

4.1 Introduction

As reviewed in chapter 3, several recent models of human associative memory rely upon the mathematical processes of convolution and correlation. These provide the single-trial associative mechanisms with which to store and recall vectors of features in a distributed memory trace (Metcalfe Eich, 1982, 1985, 1991; Lewandowsky & Murdock, 1989; Murdock, 1982, 1983, 1992, 1993). In these models, pairs of previously unseen vectors are presented simultaneously to the network. Once convolved together, the association is stored in a composite memory trace. Consequently, one of the input vectors may be selected and used to probe the memory trace so that an approximation to the second of the input vectors may be generated at the output. This process requires the reverse of convolution, correlation.

These models have proved to be very successful in modelling a wide range of human empirical data such as serial recall using TODAM (Lewandowsky & Murdock, 1989) and paired-associate learning using CHARM (Metcalfe Eich, 1982). Furthermore, when the requirement is to model traditional serial list learning and recall paradigms, their ability to perform one-shot learning has proved a significant advantage over other connectionist models that have to learn by a gradual gradient-descent learning process such as back-propagation (Rumelhart, Hinton & Williams, 1986). However, the very fact that the associative mechanisms of convolution and correlation are inbuilt within the architecture of these associative models, means that they are unable to provide a *developmental* account of associative memory.

There is much empirical data examining the development of memory in children; a comprehensive review would be beyond the scope of this thesis. Much of this work is

concerned with the development of memory span (e.g. Gathercole & Adams, 1993) and rate of articulation (e.g. Hitch, Halliday & Littler, 1993). For example, rate of articulation increases with age and has been interpreted as the cause of developmental increases in memory span (e.g. Hulme, Thomson, Muir & Lawrence, 1984). Also, the phonemic similarity effect (Conrad, 1964) becomes more apparent as the age of the subject increases (e.g. from three years to ten years; Hulme & Tordoff, 1989).

In an attempt to address the developmental nature of memory, a connectionist-like Developmental Associative Recall NETwork, DARNET, that learns-to-learn, has been developed (Brown, Hyland & Hulme, 1994; Brown, Dalloz & Hulme, 1995; Brown, Preece & Hulme, 1995; Brown, Hulme & Dalloz, 1996). DARNET uses a gradient descent learning process in order to learn how to associate novel pairs of input vectors and store them in a distributed memory trace. It can then accurately recall the second of the pair of input vectors when the first is selected randomly and presented as a probe to the memory trace. Once DARNET has learned how to form associations and retrieve items from the memory trace during this, the "phase one" stage of development, it performs single-shot associative learning at least as accurately as the convolution-correlation models described in chapter 3. Then, during "phase two" of the process, DARNET is applied to modelling some of the empirical data from studies of association including paired-associate learning (Metcalfe Eich, 1982).

In the following chapter, DARNET is introduced and both phases of the learning process described. In the first section, DARNET's architecture is examined and how it is possible for a network to "learn-to-learn" is considered. Next, we discuss the effect that the few free parameters in the DARNET architecture have on its ability to learn during this phase, before examining the impact that varying the size of the memory trace can have on the speed with which DARNET learns to learn. Finally, we examine the phase two stage and examine DARNET's ability to perform paired-associate learning, in particular with reference to empirical data presented by Metcalfe Eich (1982). We also suggest how a developmental account of cue-intrusion errors might account for the counter intuitive findings of Metcalfe Eich's data.

4.2 Phase One learning

DARNET has two distinct subnetworks: the first implements the storage of pairs of previously unseen item vectors in a memory trace and the second network implements accurate retrieval of the second of the pair when the memory trace is probed with the first. Initially all of the learned weights are assigned a small random value, so it is impossible for DARNET to accurately perform single trial learning and recall without additional training. During phase one learning, this process of storage and retrieval is optimised so that, during phase two, DARNET will be able to perform accurate single trial learning and retrieval. In this section we describe the DARNET architecture and address the question of how the network can "learn-to-learn".

4.2.1 Storage Procedure

As in convolution models such as CHARM (Metcalfe Eich, 1982, 1985, 1991) and TODAM (Lewandowsky & Murdock, 1989; Murdock, 1982, 1983, 1993) items are represented by normalised vectors, elements of which are selected randomly from a continuous normal distribution with mean of zero and variance of one.

Item vectors are connected to a layer of hidden units called "product units" by a layer of fixed connections such that the pairwise combinations of the elements from one vector and those in the second vector are represented by a unique product unit. Moreover, the product unit layer can be considered as the n-by-n square matrix that results from the matrix multiplication of the two n-dimensionality input vectors. For example, if we consider the pair of two element input vectors **a** and **b**:

$$\mathbf{a} = [a_1, a_2]$$
$$\mathbf{b} = [b_1, b_2] \qquad (4.1)$$

Then the product units take the following values:

$$p_1 = a_1b_1$$
$$p_2 = a_1b_2$$
$$p_3 = a_2b_1$$
$$p_4 = a_2b_2 \qquad (4.2)$$

This layer of product units is fully connected to the trace vector by a layer of learned weights that are modified during the phase one learning stage. This architecture is illustrated in figure 4.1.

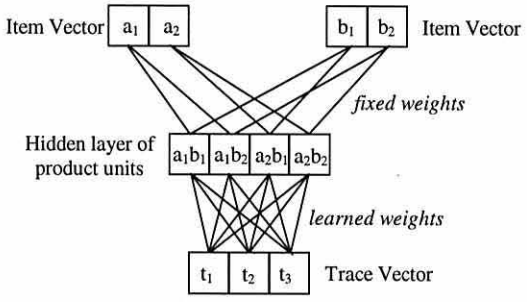

Figure 4.1 *DARNET storage architecture*

During the storage procedure, items are presented simultaneously at the input to the network and the pairwise combinations of each vector element mapped on to the hidden layer of product units. As these values are propagated to each of the trace vector elements, they are modulated by the value (or strength) of the appropriate learned weight between the product and trace vector unit. These *storage* weights are manipulated during the phase one learning in order to optimise the storage and retrieval process by minimising the error produced at the retrieval stage.

It is assumed that the input vectors, **a** and **b**, are normalised and *n*-dimensional and that the trace vector, **t**, has dimensionality *m*. It is also assumed that the learned storage weight between the product unit representing the combination of the ith element of **a** and the jth element of **b**, and the kth element of **t**, can be represented as $\mathbf{S}_{ijk}$. It is thus possible to express the storage procedure for the kth element of the trace vector as:

$$\mathbf{t}_k = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{a}_i\mathbf{b}_j\mathbf{S}_{ijk} \tag{4.3}$$

Hence it is possible for every element of the trace vector to be calculated by summing the product of each element of the hidden product layer by the appropriate storage weight connecting the elements.

4.2.2 Recall procedure

It has already been illustrated how it is possible to associate two vectors to produce a composite trace vector. During the recall procedure, one of these original input vectors is selected at random to act as a probe (i.e. a cue for retrieval) and is presented to the memory trace. In figure 4.2, item vector **a** has been selected as the probe.

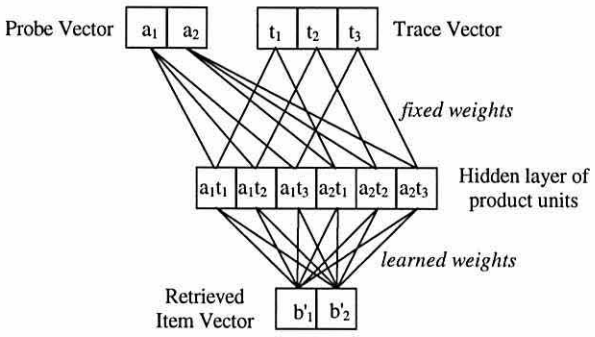

Figure 4.2 *DARNET recall architecture*

In a manner similar to the storage procedure, all the elements of both the probe item and the trace vector are connected via fixed weights to a hidden layer of product units. Once again, each element of the product unit layer represents a unique probe-trace element combination. However, in this case the product layer may be considered to be an *n*-by-*m* dimensionality matrix resultant from the matrix multiplication of the *n*-dimensionality probe vector and the *m*-dimensionality trace vector. This layer of product units is fully connected to the retrieved vector, in this case **b'**, by a layer of variable *retrieval* weights that are once again modified during the phase one learning stage.

During the recall procedure, the randomly selected probe item is presented along with the trace vector and the pairwise combinations of each probe-trace element mapped on to the hidden layer of product units by fixed strength weights. These values are then propagated to each of the retrieved vector elements and are scaled by the strength of the learned, *retrieval*, weight connecting the appropriate product and retrieved vector unit.

As before, this process may be stated mathematically. We define the *n*-dimensionality retrieved vector as **b**', and let the learned storage weight between the product unit representing the combination of the pth element of the probe vector **a**, and the kth element of the trace vector **t**, and the qth element of the retrieved vector **b**', be represented as $\mathbf{R}_{kpq}$. Therefore, the qth element of the retrieved vector may be expressed as:

$$\mathbf{b}'_q = \sum_{p=1}^{n}\sum_{k=1}^{m}\mathbf{a}_p\mathbf{t}_k\mathbf{R}_{kpq} \tag{4.4}$$

Hence it is possible to express every element in terms of either the probe and trace vectors, or by combining equation 4.4 with 4.3, in terms of the pair of input vectors by substituting for **t** in equation 4.4.

As each of the weights stored in **S** and **R** is initialised with a small random value, it is impossible for DARNET to perform single trial learning and recall of item associations accurately without additional training. This training process, the phase one learning-to-learn stage, is described in the following section.

4.2.3 Learning-to-learn

During the phase one learning-to-learn stage, the task of DARNET is to minimise the error between the retrieved vector produced by probing the memory trace with one of the input vectors, and the second of the pair of input vectors, the target vector. It achieves this by repeatedly associating novel pairs of input vectors and altering the

adjustable *storage* and *retrieval* weights, **S** and **R** respectively, in order to minimise the error between the latest target item and the retrieved item. In order to achieve this, a variant of the gradient descent algorithm (e.g. back-propagation; Rumelhart, Hinton & Williams, 1986) is employed and is presented here in an abridged form (see Brown, Hulme & Dalloz, 1996).

When each new set of input items is presented to the network and the retrieved item generated at the output, the slight change in **R**, $\Delta\mathbf{R}$, is calculated as:

$$\Delta\mathbf{R}_{kpq} = h_R\left(\mathbf{b}_q - \mathbf{b}'_q\right)\mathbf{t}_k\mathbf{a}_p \tag{4.5}$$

Where $\Delta\mathbf{R}_{kpq}$ is the small change made to the retrieval weight $\mathbf{R}_{kpq}$ during each learning iteration, h_R is the learning rate coefficient for the retrieval weights, $(\mathbf{b}_q\text{-}\mathbf{b}'_q)$ is the difference between the qth elements of the target and retrieved item vectors, $\mathbf{t}_k$ is the $\mathbf{k}^{th}$ element of the trace vector and $\mathbf{a}_p$, the pth element of the probe vector.

The corresponding learning rule for the storage weights is:

$$\Delta\mathbf{S}_{ijk} = h_S\mathbf{a}_i\mathbf{b}_j\sum_{q=1}^{n}\left(\left(\mathbf{b}_q - \mathbf{b}'_q\right)\sum_{p=1}^{n}\mathbf{a}_p\mathbf{R}_{kpq}\right) \tag{4.6}$$

where $\Delta\mathbf{S}_{ijk}$ is the small change made to the storage weight $\mathbf{S}_{ijk}$ during each learning iteration, h_S is the learning rate coefficient for the storage weights (and is typically equal to h_R and take a small value of approximately 0.1), $(\mathbf{b}_q\text{-}\mathbf{b}'_q)$ is again the difference between the qth elements of the target and retrieved item vectors and $\mathbf{a}_i$ and $\mathbf{b}_j$ are the ith and j^{th} elements of the input vectors, **a** and **b**. Also, $\mathbf{a}_p$, is the pth element of the probe vector and finally, $\mathbf{R}_{kpq}$, is the retrieval weight.

This iterative process of presenting previously unseen items to the network, then changing both the storage and retrieval weights in order to minimise the error at the output, is repeated many times with novel pairs of items until DARNET has learned to accurately store and retrieve pairs of items from the composite memory trace. Clearly

the ability to learn and recall novel pairs of items in a single trial contrasts with traditional back-propagation models which learn a specific set of data by repeated presention during training.

To summarise the phase one learning stage: novel items, represented by normalised n-dimensionality vectors, are presented in pairs to the network and the association between them, formed by propagating the product layer representation of the vectors by the learned storage weights, stored in the m-dimensionality memory trace. During this learning-to-learn process, DARNET must alter the (learned) storage and retrieval weights that link the hidden layer of product units with either the memory trace or the retrieved item, by small amounts so as to minimise the error between the target item and the retrieved item. Once trained to a predetermined degree of accuracy, DARNET should be able to accurately store and recall previously unseen items to and from the memory trace in a single trial.

In the next section, simulations are presented which illustrate how DARNET learns to learn, along with an investigation of the effects of manipulating the limited number of free parameters in the DARNET architecture.

4.3 Simulations during the phase one learning stage

4.3.1 Simulation 1: *Learning-to-learn*

Introduction

In the following simulation, the results of training the network to perform the item storage and retrieval described in section 4.2 are presented. The model was required to learn to associate novel pairs of items to a sufficient degree of accuracy such that the model could be shown to perform near perfect item storage and retrieval. The aim of this simulation was to replicate earlier results (e.g. Brown, Hulme & Dalloz, 1996) which illustrate how DARNET learns to learn.

Method

For this simulation, DARNET was presented with novel pairs of ten element item vectors. Each element was selected randomly from a continuous normal distribution of scalars with mean zero and variance of one. Each item vector was normalised in order that the dot product between any one item vector and *itself* was unity. The association formed after presentation was stored in a memory trace of dimensionality 19 elements. (This trace vector dimensionality was selected in order to ensure that this latest implementation of the DARNET architecture could replicate earlier results presented in Brown, Hulme & Dalloz (1996) in which DARNET was trained to perform convolution where the dimensionality of the memory trace is required to be $2n$-1 for n-dimensionality input vectors). The storage and retrieval weights in network were initialised with small random values with mean of zero.

Before each new learning epoch, each of the 19 memory trace elements was reset to zero and a novel pair of items generated at the input. These input vectors were then propagated via the hidden layer of product units and modulated by the storage weights in order to generate a memory trace vector.

Next, one of the input vectors was selected at random and presented as a probe, along with the memory trace, to the second hidden layer of product units. The pattern of activation across these product units was then modulated by the retrieval weights and an approximation to the target item generated at the output.

The error score between the target item and the approximation was calculated. Finally, the learning rules (equations 4.5 and 4.6) were applied to the storage and retrieval weights in order to minimise the error at the output. If the error at the output was found to fall below a specific criterion, phase one training ceased, otherwise a new epoch began and the process described here was repeated once more.

Results

Figure 4.3 illustrates how the mean squared error between the retrieved item and the target item decreases as the number of training epochs increases. The mean-squared

error score is calculated by summing the squared difference between each element of the target item and the corresponding element of the retrieved vector. However, if this error score is plotted alone, the learning curve generated can be erratic even though the overall trend is towards a decreasing error score (e.g. Brown, Hulme & Dalloz, 1996, figure 5). This problem is overcome by averaging the error after each (set of) epoch(s).

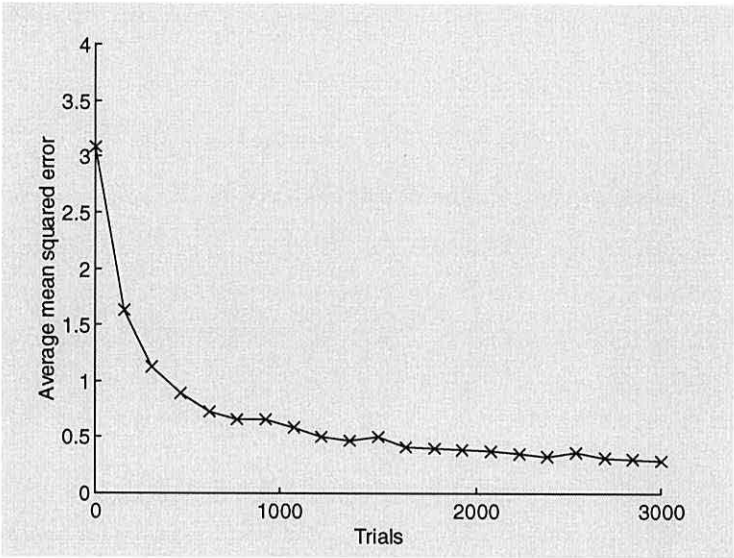

Figure 4.3 *Error score between the output and the target item as DARNET learns-to-learn*

More precisely: in order to generate the smooth learning curve of figure 4.3, at the point when the error score was to be calculated, phase one learning was suspended and instead a number of novel input vectors were presented and the mean-squared error calculated as described above. However, this mean-squared error score was then averaged for a number of different input vector pairs and the *average* mean-square error calculated. The net result is that plotting the average mean-squared error score gives a more accurate indication of the performance of the network at any one time.

It is clear from figure 4.3 that DARNET learns to perform accurate item storage and retrieval after three thousand epochs of phase one learning.

Discussion

Comparison of the results presented here with data presented in Brown, Hulme and Dalloz (1996, figure 5) confirms that DARNET is in fact performing to a similar level of performance as would be expected from a convolution-correlation model. Brown, Hulme & Dalloz (1996) demonstrate that it is possible to train DARNET to perform convolution and correlation. Analysis of the learned weights and memory trace reveals that, although DARNET may use the same dimensionality memory trace vector as a convolution model, and a similar set of stimuli, DARNET does not use convolution as a default associative mechanism unless specifically trained to do so.

In summary, we can confirm that DARNET can learn-to-learn. By applying a gradient descent based algorithm to the storage and retrieval weights, DARNET can learn to perform single-shot association and recall of novel pairs of item vectors to a degree of accuracy similar to that obtained by convolution-correlation based models such as CHARM (Metcalfe Eich, 1982, 1985, 1991) and TODAM (Lewandowsky & Murdock, 1989; Murdock, 1982, 1983, 1992, 1993).

However, DARNET possesses a number of free parameters that can affect its ability to perform phase one learning. In the following simulation, the first of these parameters, the *learning rate* parameter, is manipulated and the effect on learning investigated. This represents the first in a series of simulations that explores the basic computational properties of the architecture prior to addressing the psychological data.

4.3.2 Simulation 2: *Effect of varying the learning rate*

Introduction

Equations 4.5 and 4.6, the storage and retrieval weight change algorithms, include the learning rate coefficients, h_S and h_R. These determine the strength with which the error between the retrieved vector and the target vector is propagated across the weights in the network. Typically these take a small value in order to ensure that the learning process is gradual and less erratic than if they were greater. However, if the learning rate is increased slightly, it is anticipated that the network should learn-to-learn to a

similar degree of accuracy as the network in the previous section, but after a shorter period of training.

In the following simulation, a similar network to that used before is presented. However, in this case the learning rate parameter is manipulated and the affect on phase one learning performance investigated.

Method

For the following simulation, the same method as that described in section 4.3.1.2 was employed. However, in this case, the storage delta weight and retrieval delta weight *learning rate* parameters, h_S and h_R respectively, were varied. However, in all cases h_S was equal to h_R.

Results

Figure 4.4 illustrates the average mean-squared error decreasing over two sets of 20,000 epochs of phase one learning. In the first condition, the learning rate is 0.05 while in the second it is assigned a value of 0.10.

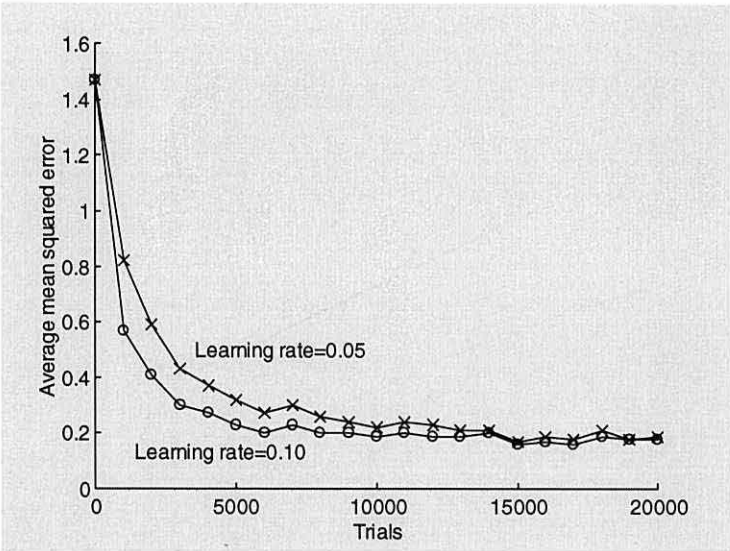

Figure 4.4 *Effect of increased learning rate during phase one training*

It is clear from these results that when the learning rate parameter is increased, DARNET reaches the asymptotic level of performance after approximately six

thousand trials. At this point, the condition with the lower learning rate is still performing slightly worse and does not in fact reach the same degree of performance until after a further ten thousand epochs of training.

Discussion

It is clear from the findings presented in figure 4.4 that judicious use of a learning rate coefficient can reduce the amount of time taken to train the network to perform accurate item association and retrieval. However, it is also clear from these results that the use of a learning rate parameter will not improve the overall performance of the network. It will only reduce the amount of time required by the network to train to that same level of performance.

In the following section, the second of the free parameters, the momentum term, is manipulated and the impact that it has on the performance of the network during the phase one stage assessed.

4.3.3 Simulation 3: *Effect of varying the momentum parameter*

Introduction

In the following simulation, a *momentum* coefficient was introduced to the DARNET architecture in order to improve performance during the phase one learning stage. Before examining the influence of this parameter, it is important to explain why this term should be introduced to the network.

Consider the common analogy used when describing neural network learning, that of the path followed by a ball rolling down a slope. If you assume that a ball at the top of a hill represents the state of the network immediately before the first epoch, the base of the hill as representing the state of the network when it is performing error free association and retrieval, then the displacement of the ball at any time from the base corresponds to the error in the network. So, the path followed by the DARNET network during the phase one learning stage can be compared to the path taken by the

ball, rolling from the initial state of maximum error to its final resting position when the network is considered to have learned how to perform accurately.

During the phase one stage, when the first set of items has been presented to the network and the error between the item at the output and the target item calculated, the *first* set of delta-weights can be generated. These weights act to minimise the error between the current output and the target item. As this is the first time that any delta-weights have been calculated, we anticipate that they will produce the greatest change in the network as it should be in a state of maximum error. Referring back to the ball-on-slope analogy, as these weight changes are applied to the network, so we can imagine that the ball has become agitated and has started to travel in the *steepest* possible direction from the summit towards the base of the hill. Clearly, the ball acts to minimise its displacement (i.e. error) above the baseline as quickly as possible.

Without a momentum term, the network recalculates the delta-weights again during the next epoch without any consideration of those calculated during the previous trial. Referring again to the ball-on-slope analogy, without momentum the ball will stop travelling before the next epoch. Its displacement above the base will be re-evaluated and a second set of delta weights calculated. Although this new velocity may not be in as steep a direction as during the first epoch, no consideration is made of the direction in which the ball was travelling prior to this latest epoch. The new velocity is based solely upon the current state of the network. However, the introduction of a momentum term ensures that before the new velocity is calculated, the direction in which the ball was travelling previously is taken into consideration.

This may be clarified by considering the following equation in which the momentum term, M_R, is shown to scale the previous delta-weights before they are combined with the current state of the network to produce the next set of delta-weights:

$$\Delta\mathbf{R}(t+1)_{kpq} = h_R\left(\mathbf{b}(t)_q - \mathbf{b}'(t)_q\right)\mathbf{t}(t)_k\mathbf{a}(t)_p + \mathbf{M}_R\Delta\mathbf{R}(t-1)_{kpq} \qquad (4.7)$$

As before, $\Delta\mathbf{R}_{kpq}$ is the small change made to the retrieval weight $\mathbf{R}_{kpq}$ during each learning iteration. The corresponding learning rule for the storage weights is:

$$\Delta\mathbf{S}(t+1)_{ijk} = h_S\mathbf{a}(t)_i\mathbf{b}(t)_j\sum_{q=1}^{n}\left(\left(\mathbf{b}(t)_q - \mathbf{b}'(t)_q\right)\sum_{p=1}^{n}\mathbf{a}(t)_p\mathbf{R}(t)_{kpq}\right) + M_S\Delta\mathbf{S}(t-1)_{ijk} \quad (4.8)$$

Where $\Delta\mathbf{S}_{ijk}$ is the small change made to the storage weight $\mathbf{S}_{ijk}$ during each learning iteration and M_S the momentum term for the storage weights.

The net result of the inclusion of a momentum term is that the path from maximum error towards the error free state should be shorter. Hence the network should learn over a shorter number of epochs. Therefore, in the following simulation, a momentum term is introduced to both the storage and retrieval delta-weights calculation. In the data presented here, M_S is equal to M_R.

Method

The method employed during the following simulation is identical to that described in the first simulation (section 4.3.1.2). However, the addition of a *momentum* term means that rather than resetting the delta weights, $\Delta\mathbf{R}_{kpq}$ and $\Delta\mathbf{S}_{ijk}$, to zero before making the changes to the storage and retrieval weights, in accordance with equations 4.7 and 4.8, a proportion of the previous delta-weights is combined with the calculation of the current delta-weights before all of the weights were updated.

Results

The suggestion that a momentum term will improve the ability of the network to learn is largely confirmed by figure 4.5 where it is clear that the introduction of a large momentum term results in the network learning to associate and retrieve items with a high degree of accuracy after only two thousand epochs of phase one training.

However, it is also clear that in this case, the network has suffered as the overall level of performance is reduced compared to that obtained by the previous, momentum free, method.

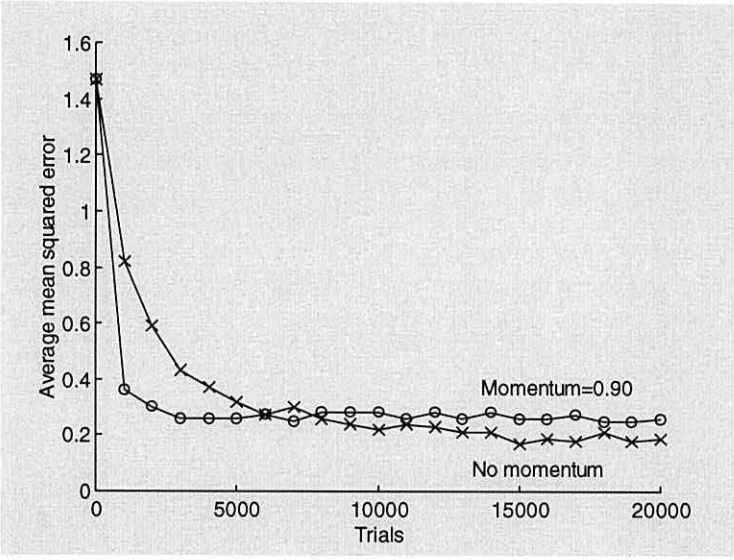

Figure 4.5 *Effect of momentum term during phase one training*

Discussion

It is evident from figure 4.5 that the inclusion of a momentum term can improve the rate at which DARNET learns-to-learn during the phase one training stage. However, it is clear that in this case, the reduction in phase one training time is at the expense of overall accuracy.

In fact, it is probable that the network using the momentum term has become lodged in a deep local minimum in the error space. Returning to the ball-on-slope analogy once more, you could imagine that the ball has become lodged in a ravine somewhere on the descent. Hence the only way to make further progress towards the base, is to first climb back out of the ravine before resuming the downhill track once more. If the training were continued beyond the twenty thousand epochs presented here, the mean squared error would be expected to drop further, settling at a similar value to that exhibited by the momentum free version (see Haykin, 1994).

However it is not just the additional momentum and learning rate parameters that effect DARNET's performance. The most dramatic effect results from varying the dimensionality of the memory trace vector. In all of the simulations presented so far,

this has been held constant at 19 elements for a 10 element input vector. In the next section, we examine the effect of increasing the dimensionality of the trace vector.

4.3.4 Simulation 4: *Trace dimensionality effects*

Introduction

Convolution-correlation based models of association are forced to use a memory trace vector of dimensionality $2n$-1 for n-dimensionality input vectors. This is one of the inherent requirements of the convolution process (see Metcalfe Eich, 1982). However, although we have illustrated in the simulations described thus far that DARNET will train the storage and retrieval weights to perform single-shot associative learning and recall with a memory trace whose dimensionality is calculated in a similar fashion, this need not be the case.

DARNET, in contrast, does not rely on the trace being a specific dimensionality in order to perform item storage and recall adequately. For example, at one extreme it is possible to envisage a trace vector of dimensionality n^2, in which case DARNET could potentially recognise that the memory trace could contain one-to-one mappings of each of the hidden product units. We anticipate that recall should be near perfect and error free. At the other extreme, if the memory trace was only one element long, we could anticipate that DARNET might find storing and retrieving the association of a pair of 10 element input vectors impossible.

Therefore, in the following simulation, we investigate the range of performance from DARNET when the trace dimensionality is varied within these bounds. Brown, Hyland and Hulme (1994) suggested that there an optimal trace dimensionality for DARNET is $0.5n(n+1)$. Here, we aim to replicate this finding.

Method

For the following simulation, the same method as was described in section 4.3.1.2 for simulation 1 is employed. However, unlike the previous DARNET simulations, in the present simulation the dimensionality of the trace vector was varied for each of the five

conditions. In the first condition, it was assigned a value of four elements, equal to the dimensionality of the stimuli. Subsequent conditions employed dimensionalities of seven, ten, twelve and fourteen elements. In all cases, the network was trained over ten thousand phase one epochs.

Results

Figure 4.6 illustrates the result of plotting the average mean squared error over the ten thousand trials. Note that in this case, it is the *log* of the mean squared error that is reported as it provides a clearer indication of how the error is reducing over the later training epochs. The results clearly demonstrate that the dimensionality of the trace vector can have a huge effect on DARNET's ability to perform accurate single-shot associative learning.

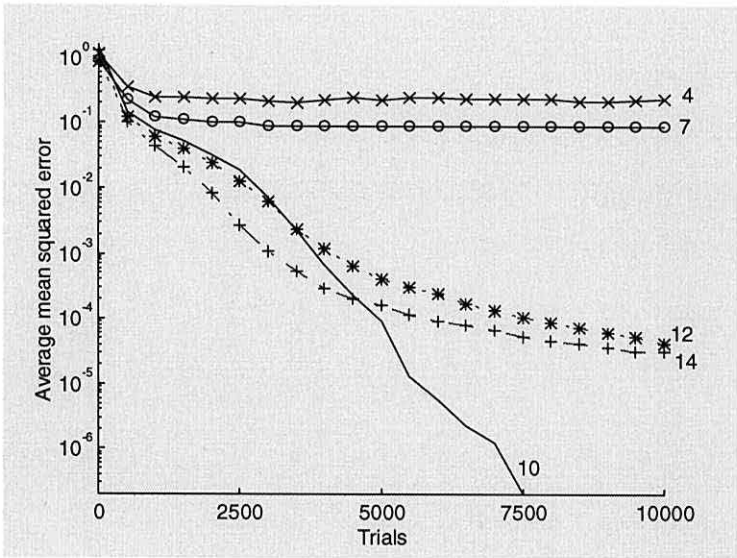

Figure 4.6 *Effect of increasing the trace length from 4 to 14 elements on the capacity to store and recall four element input vectors*

When the trace has the same dimensionality as the item vectors (n=4) performance is clearly poor as the error reaches an asymptotic level after only two thousand epochs. The same is also true when the trace has the same dimensionality as if it were a convolution-correlation model ($2n$-1=7). However, as the dimensionality increases further, so the network takes advantage of the extra capacity of the memory trace and performance improves in all of the remaining cases. The model does not settle at an asymptotic level of performance even after the ten thousand epochs of phase one

learning. However, what is immediately clear is that the largest capacity memory trace does not guarantee the superior performance. In fact, it is when the memory trace has dimensionality 10 that the network performs with the highest degree of accuracy.

Discussion

As is evident from figure 4.6, performance is poor (but probably adequate) in the first two cases where the trace dimensionality is either identical to the n-dimensionality of the input vectors or the $2n$-1 dimensionality of a convolution-based memory trace vector. In fact, this is significant as DARNET has already been shown to perform to a similar level of performance as convolution-correlation models when given a memory trace with dimensionality equal to that of the corresponding convolution-correlation associative network. The results presented in figure 4.6 suggest that performance in this condition is poor in comparison with that which might be attained if DARNET were trained using a larger capacity memory trace.

Increasing the trace dimensionality to either 12 or 14 elements results in improved performance. However, although the error has not reached an asymptotic plateau after ten thousand epochs of phase one training, DARNET performs near perfect item storage and retrieval during a phase two learning task. In fact, the average mean squared error at these points is almost one thousandth of that of the network with the trace dimensionality $2n$-1.

However, the most startling result from this simulation using four element input vectors occurs when the memory trace vector has a dimensionality of 10 elements. In this case performance can be seen to fall between the two previous sets of results over the first 2500 epochs, however, around this point during the phase one training, the network appears to suddenly recognise how to store the associations accurately. As a result, the learning curve falls rapidly towards zero error. After 8000 epochs, the average mean squared error is approximately 0.0000001. This finding, that the optimal DARNET memory trace size for n-dimensionality input vectors is $0.5n(n+1)$, is confirmed by similar results using three element input vectors where the optimal trace

dimensionality is six elements. A similar result is presented in Brown, Hyland and Hulme (1994), however they made no attempt to explain it.

Mathematical analysis of the weights arrays after phase one learning reveals that when given a trace with dimensionality $0.5n(n+1)$ elements, DARNET discovers a symmetric relationship between some of the storage and retrieval weights. Examination of the storage weights in table 4.1 reveals that $\mathbf{S}_{11k}= -\mathbf{S}_{22k}$. More specifically, the storage weights connecting every element of the trace vector to the hidden layer unit containing the product of first element of **a** and the first element of **b** are equal in magnitude, but of a different sign, to the storage weights connecting the corresponding elements of the trace vector to the hidden layer unit containing the product of the second element of **a** and the second element of **b**. A similar relationship links the product units containing the product of the first element of **a** and the second element of **b**, and that of the second element of **a** and the first element of **b**, except in this case the weights in both cases are identical for the corresponding trace vector elements: $\mathbf{S}_{12k}= \mathbf{S}_{21k}$.

Table 4.1

Storage weights, S_{ijk}, after training to high degree of accuracy using a trace length of 0.5n(n+1)

	k		
	1	2	3
i=1			
j=1	-0.4280	0.3159	0.8509
j=2	-0.2509	-0.9308	0.1336
i=2			
j=1	-0.2509	-0.9308	0.1336
j=2	0.4282	-0.3160	-0.8508

However, the choice of $0.5n(n+1)$ for the memory trace dimensionality may not in fact be desirable as this will increase the number of weights needing to be learned during the phase one stage. Although this may not be of concern given the small dimensionality stimuli used in this simulation, if larger (e.g. n=32) item vectors are required, the dimensionality of the memory trace may have to be reduced in order to allow adequate phase one training given limited time and computing resources.

4.3.5 Summary

From the findings of the four simulations presented here, it is evident that DARNET can learn to associate pairs of novel input vectors, store them in a memory trace and subsequently retrieve one of the stimuli when the other is presented as a recall cue.

It has also been shown that the introduction of both a learning rate and momentum term can influence the phase one learning. However, care must be taken in order to ensure that the overall level of performance does not suffer as a result of the reduction in time spent during phase one learning. Furthermore, it is clear that the dimensionality of the memory trace vector has a huge influence on the ability of DARNET to perform accurately. We suggest that for n-dimensional stimuli, a memory trace of dimensionality $0.5n(n+1)$ elements will provide optimal performance. However, the choice of memory trace dimensionality may be influenced by the resources available during the phase one training.

It should also be clear that the gradual learning process employed by DARNET during the phase one stage means that DARNET can provide a developmental account of single-trial learning. In the following section, the performance of DARNET during phase two learning and recall is addressed.

4.4 Phase Two learning

In the previous section, the ability of DARNET to learn-to-learn during the phase one stage was described and results were presented that confirmed that DARNET can learn to store the association of previously unseen pairs of normalised vectors in a single trial, and accurately recall one of them given the other as a recall cue.

In the present section, the ability of DARNET to perform paired-associate learning is examined. Firstly, the effect of increasing the trace dimensionality on the network's capacity for learning associations is considered for both confusable and nonconfusable stimuli. Closer analysis of DARNET's performance for confusable and nonconfusable

paired-associate learning and recall follows and includes a comparison with results from simulations using CHARM (Metcalfe Eich, 1982, simulation 1).

In the final section DARNET is once again required to perform paired-associate learning and attention is focused in particular on a Metcalfe Eich prediction based upon simulations using CHARM. She suggests that subjects may exhibit a high proportion of cue intrusion errors during paired-associate recall of confusable stimuli. Metcalfe Eich's initial empirical findings (Metcalfe Eich, 1982, experiment 1) fail to support this prediction, however a refined experiment confirms them (Metcalfe Eich, 1982, experiment 2). DARNET reproduces this finding and reveals that an equally high proportion of intrusion errors occur when a *partially trained* network attempts paired-associate learning and recall. However, the effect is minimal for a highly trained network, providing support for the suggestion that a developmental account of associative memory is required (Brown, Preece & Hulme, 1995).

4.4.1 Simulation 5: *Trace dimensionality effects*

Introduction

For each of the phase two simulations, DARNET implements paired-associate learning and recall in accordance with the method outlined by Metcalfe Eich (1982). Using CHARM, Metcalfe Eich randomly selected six items, each represented as a 63-element vector, from a vocabulary of 12 items and presented them in pairs to the network. Association was achieved by the mathematical process of convolution and each memory trace that was generated, was accumulated in a composite trace, given by the equation:

$$\mathbf{t} = \mathbf{a}*\mathbf{b} + \mathbf{c}*\mathbf{d} + \mathbf{e}*\mathbf{f} \qquad (4.9)$$

In equation 4.9, items are represented by the vectors **a** through **f**, and the convolution of the first pair of vectors, **a** and **b**, by the notation **a*b**. Importantly, CHARM's composite memory trace was truncated to the central 63-elements.

During recall, the first, third and fifth items were used to probe the composite memory trace. This process involved correlating each probe item with the memory trace in order to generate approximations to the second, fourth and sixth items at the output. The retrieved items were once again truncated to the central 63 features. The retrieved patterns were then compared with the vocabulary of 12 recallable items and the closest match, measured by summing the product of each element of the retrieved vector with the corresponding element of the vocabulary item, (the *resonance score* or inner product) recalled as the network's response. Metcalfe Eich repeated the simulation using 50% similar, confusable, items. These were generated by copying every other feature from the first item to each of the first six items.

However, it must be noted that Metcalfe Eich confuses the issue of whether or not the similar stimuli were still normalised once the process of introducing similarity had been completed (Metcalfe Eich, 1982, p. 637). This is potentially very significant as the recognition process Metcalfe Eich describes relies on the stimuli being normalised in order that the dot product provide a suitable means for distinguishing between responses. For each of the DARNET simulations described here, items that are similar are also normalised. The process for introducing similarity without destroying the normalisation is described in detail in the appendix.

In analysing the results, Metcalfe Eich classified each response into one of five response categories: *target* (e.g. item **b** if item **a** was presented); *cue* (e.g. item **a** if item **a** was presented); *nontarget response* (e.g. item **d** or **f** if item **a** was presented); *nontarget cue* (e.g. item **c** or **e** if item **a** was presented); or *unrelated extralist* (e.g. **g**, **h**, **i**, **j**, **k** or **l**, the distractor items completing the vocabulary).

In this first phase two simulation, the effect of increasing the capacity of the memory trace during a phase two simulation is considered (see Brown, Hyland & Hulme, 1994).

Method

DARNET was trained to an asymptotic level (or for 20 000 epochs if an asymptote has not been achieved) during the phase one training stage using eight element stimuli in order to reduce the amount of time required to learn. The dimensionality of the composite memory trace was manipulated, being assigned values of either 8, 16 or 32 elements. The state of the storage and retrieval weights once learning was complete was preserved by storing each set of weights so that further phase one training was unnecessary when replicating the simulation.

During phase two, DARNET was presented with three pairs of normalised eight-element input vectors with elements drawn from a continuous normal sample with mean of zero and variance of one. In a manner similar to Metcalfe Eich (1982), three pairs of nonconfusable stimuli selected from a lexicon of 12, were presented sequentially to the network. Memory traces were calculated for each pair and accumulated in a composite memory trace. During recall, the first of each of the item pairs was presented to the network as a probe and the output compared with the lexicon of 12 recallable items. In the second condition, confusable stimuli were presented to the network. Using a normalising technique to ensure both 50% similarity and item normalisation, six confusable items were generated and presented to the network. Six nonconfusable distractor items were added to the lexicon of recallable items.

By comparing the retrieved item in each case with the lexicon of allowed responses, and classifying the responses appropriately, it is possible to record the mean proportion of correctly recalled items, for each trace dimensionality condition, for both the nonconfusable and confusable list conditions.

Results

The mean proportion of correct responses for both nonconfusable and confusable items are presented in figure 4.7. For the nonconfusable condition, performance increases from 52% when the trace dimension is equal to that of the item vectors, to 77% when the trace dimensionality is three times that of the stimuli. Likewise, for the

50% similar (confusable) items, performance increases steadily with the trace size, from 21% to approximately 39%.

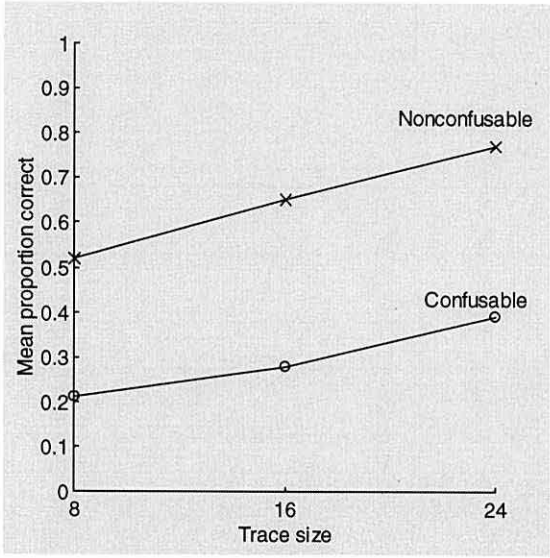

Figure 4.7 *Effect of increasing the dimensionality of the composite memory trace during a phase two paired-associate learning task for both nonconfusable and confusable stimuli*

Discussion

The results summarised in figure 4.7 illustrate that increasing the capacity of the memory trace vector will result in improved phase two learning and recall performance. It is also apparent from the gradient of each plot that the degree of improvement for both the nonconfusable and confusable stimuli is similar. Given these findings, we can suggest that any overall reduction in performance exhibited by DARNET may be attributable to the limited dimensionality and capacity of the memory trace vector.

4.4.2 Simulation 6: *Paired-associate intralist intrusions*

Introduction

The aim of the following simulation was to attempt to replicate the distribution of errors found by Metcalfe Eich (1982) using CHARM. Metcalfe Eich observed that CHARM scored highly when recalling nonconfusable items, but much poorer for 50% confusable items. This simulation also aims to replicate some of the findings of Brown, Dalloz and Hulme (1995, simulation 1).

Metcalfe Eich reported that CHARM performed better with unrelated (nonconfusable) items, recalling 99% of the items correctly in the unrelated condition compared with only 42% in the similar (confusable) condition. Furthermore, she observed that nontarget responses and nontarget stimulus scores were both identical, with 22% of responses in the similar condition. Also, in both conditions almost none of the unrelated extralist items were retrieved. These results are summarised in table 4.2.

Table 4.2

Mean percentage recall for CHARM (adapted from Metcalfe Eich, 1982, table 1)

Condition	Target	Cue	Nontarget response	Nontarget stimulus	Unrelated extralist
Unrelated	98.6%	0.6%	0.6%	0.0%	0.3%
Similar	42.0%	14.6%	21.7%	21.7%	0.0%

CHARM's predictions are largely confirmed experimentally by Metcalfe Eich (1982, experiment 1). She reports that human subjects also produce many more errors in the confusable (related or categorized) condition than in the nonconfusable (unrelated or uncategorized) condition: 80% versus 56% appropriately (cf. Conrad, 1965). However, Metcalfe Eich observed that in contrast to the prediction made by CHARM, no human subjects produced cue intrusion errors, which she attributes to the subject's use of a high-level strategy preventing the recall of a cue item as a response. This last issue is addressed in detail in the next section.

Method

The procedure for this simulation is identical to that for the previous simulation, except that the trace dimensionality remained constant. Also, Metcalfe Eich's simulation used 63 element input and memory trace vectors. However, because DARNET, in contrast to CHARM, requires phase one learning before it can be applied to a phase two task, the time required to train a DARNET model to a similar capacity was considered to be too great. Therefore, DARNET was trained for input vectors of dimensionality 10 and a trace vector of dimensionality 19 elements.

Results

The results, presented in table 4.3, illustrate that DARNET performs adequately given the limitations of the dimensionality of both item and trace vectors.

Table 4.3

Mean percentage recall for DARNET replication of Metcalfe Eich's Simulation 1 (1982)

Condition	Target	Cue	Nontarget response	Nontarget stimulus	Unrelated extralist
Unrelated	69%	5%	7%	4%	15%
Related	39%	3%	30%	27%	2%

Using unrelated stimuli, DARNET scores 69% correct and only 5% cue intrusion errors. Both nontarget responses and nontarget stimuli register low scores (7% and 4% respectively). However, 15% of responses are classified as unrelated extralist responses.

In contrast, when the 50% similar, confusable stimuli are used, DARNET's performance is reduced for every category (39% are target response, 30% as nontarget responses and 27% as nontarget stimuli responses) except for cue intrusions and unrelated extralist responses. In the former, the percentage decreases slightly to 3% while in the latter, the percentage decreases from 15% to 2%.

Discussion

The results presented in table 4.3 compare favourably with both those of CHARM (Metcalfe Eich, 1982) and DARNET (Brown, Dalloz & Hulme, 1995). However, although the overall form of these results suggests that DARNET is performing paired-associate recall in a similar fashion to CHARM, there are three discrepancies between the results presented here and those of Metcalfe Eich.

The first is that overall performance is reduced in comparison to that of CHARM: DARNET scores only 69% of responses as correct compared to CHARM's 99%. This reduction in accuracy may be accounted for by considering both the degree of phase one learning and the capacity (i.e. dimensionality) of the memory trace in comparison with that of the input vectors. As was stated previously, for this simulation phase one

training was halted at an asymptotic level. Recalling figure 4.6, it is clear that for a network using a trace dimensionality of $2n$-1 elements, the performance reaches an asymptote quickly and further amounts of phase one learning are not rewarded with an improvement in performance. It is also highly likely that the limited dimensionality of the memory trace vector will have had a marked effect on overall performance. Simulation 5 demonstrated that increasing the dimensionality of the memory trace vector improves overall performance for both nonconfusable and confusable stimuli.

The second difference between the DARNET data and the CHARM data is the increased number of unrelated extralist intrusion errors (15% compared with 0.3% for unrelated items). As before, this level of performance can be attributed to the inability of the network to accurately retrieve encoded items. This suggests, once again, that either the network is not sufficiently well trained to encode and decode items from the memory trace, or that the memory trace lacks the capacity to allow items to be stored accurately.

The third observation is that there are many fewer cue intrusion errors, particularly in the case of the similar items, when compared with CHARM. In fact, although the performance of the network has been deemed poor for the unrelated condition when compared with that of CHARM, it appears that performance for similar items is in fact slightly better. In an effort to account for the high proportion of cue intrusions in the CHARM data, we will next consider DARNET's performance *midway* through the phase one training: this amounts to a developmental analysis of paired-associate learning.

4.4.3 Simulation 7: *A developmental account for cue intrusions*

Introduction

If one recalls the shape of the learning curve produced by recording the average mean-squared error during the phase one learning stage (figure 4.3), it is clear that if it was possible to capture the state of the model at intervals over the lifetime of this phase one training, it might be possible to examine the developmental nature of DARNET's

performance. By recording the state of the storage and retrieval weights at discrete intervals over the duration of the phase one learning stage, and reinstating each of these weight states during the phase two learning and recall procedures, it is possible to examine the developmental nature of phase two learning and recall.

Metcalfe Eich (1982), in the simulation reported in section 4.4.2, observed a high proportion of cue intrusions when using the 50% similar items. She suggested that her finding, that almost 15% of responses (25% of all errors) are cue intrusions, is "counterintuitive" (Metcalfe Eich, 1982, p. 638) and in a follow up study (Metcalfe Eich, 1982, experiment 1) which found that subjects do not produce cue intrusion errors, suggested that this may be due to the presence of a rule to eliminate such responses. However, in a second experiment (Metcalfe Eich, 1982, experiment 2), she reports that a higher proportion of cue intrusion errors occur in a confusable *synonym* condition than occur in an unrelated condition (55% versus 39% in the unrelated condition), a finding in line with CHARM's prediction.

Therefore, in the following simulation, DARNET was once again applied to the problem of paired-associate learning and recall, however in this case the ability of DARNET to perform these tasks at two different stages in its phase one development was addressed.

Method

In a replication of the procedure described in section 4.4.1.2, DARNET was presented with three 10 element paired-associates which it accumulated in a composite memory trace with dimensionality of 19 elements. In this case, items were selected from a vocabulary consisting of six 50% similar normalised items. A further six unrelated normalised items were added as distractor items during recall.

However, DARNET employed the use of two different sets of storage and retrieval weights: a *partially trained* set, generated midway through the phase one learning stage when the mean-squared error in the network was around 50%; and a *well*

trained set, trained to a level of around 10% mean-squared error (identical to those used in the previous simulation).

Once again, responses were categorised in the manner described by Metcalfe Eich (1982). However, in order to analyse the developmental nature of the errors, the proportion of errors which are cue intrusion errors was also recorded.

Results

The results for the partially and well-trained networks are summarised in table 4.4. It is clear that when compared with the performance of the well learned network, although there is a lower level of overall performance (27% correct compared with 39% correct), it is the cue intrusion errors that increase by the greatest extent (from only 3% to 16%). The proportion of nontarget, nontarget stimulus and unrelated extralist errors remain similar for both conditions. Nontarget responses decrease by 5%, nontarget stimulus errors increase slightly by 2% while unrelated extra list errors increase by 6%.

Table 4.4

Mean percentage recall for confusable items with well trained and partially trained DARNET

Condition	Target	Cue	Nontarget response	Nontarget stimulus	Unrelated extralist
Well trained	39%	3%	30%	27%	2%
Partially trained	27%	16%	25%	25%	8%

However, closer examination of the proportion of errors which are cue intrusions reveals the full extent to which they increase. In the well trained condition, cue intrusions provide only 5% of the errors exhibited by the network. However, this increases to 22% of all the errors in the partially trained condition.

Discussion

The results of the current simulation clearly demonstrate that a development account of paired-associate learning can provide a possible account for the distribution of errors occurring in the similar item condition.

It is clear that the level of associative ability exhibited by the network influences the type of error produced. The results also suggest that the proportion of cue intrusion errors produced by the network decreases as the ability to perform accurate association improves. These data suggest therefore, that human subjects might also exhibit a developmental distribution of error types. For example, the number of cue intrusions may decrease towards zero as the subjects develop (which accounts for Metcalfe Eich not finding any cue intrusions when testing her college-age subjects).

Also, these results reveal that a particular type of error may be due either to the accuracy of the storage and retrieval mechanism employed by a model *or* the complexity of the task being undertaken. In the present case, it appears that higher numbers of cue intrusion errors could occur as a result of either the confusability between items or the associative ability of the network.

It is possible to conclude from these results that a high proportion of errors, such as the cue intrusion errors reported by Metcalfe Eich in her CHARM simulation using 50% similar items (Metcalfe Eich, 1982), may be reproduced by DARNET at an early stage in its development but eliminated at a later stage. And as such, Metcalfe Eich's results may reflect the inability of CHARM to perform accurately when presented with complex stimuli such as the 50% similar items.

4.5 Summary

In this chapter, a developmental associative recall network, DARNET, has been introduced. DARNET, a multi-layer gradient descent based network has been shown to learn how to form accurate single-trial association and recall during the phase one stage. The few free parameters that effect the ability of the network, the momentum and learning rate parameters, have been introduced and the effect of varying each parameter during phase one learning has been reported. The effect of varying the memory trace dimensionality demonstrated that for n-dimensionality input vectors, DARNET has an optimal memory trace size of $0.5n(n+1)$ elements.

Once DARNET has learned how to perform single shot item learning and retrieval to a level of performance similar to the convolution and correlation models of association (e.g. CHARM: Metcalfe Eich, 1982, 1985, 1991; TODAM: Murdock, 1982, 1983, 1992, 1993), it can be applied to replicating data modelled by other such models such as the problem of paired-associate learning.

Finally, the significance of a developmental account of learning was highlighted by examining paired-associate recall error data reported by Metcalfe Eich (1982) using CHARM. In particular, the proportion of cue intrusion errors that occurred when using three pairs of 50% similar items as the stimuli. In a replication of her simulation, DARNET was found to produce a similar proportion of cue intrusion errors in the case when the network had only been partially trained during the phase one stage. The cue intrusion errors were found to be almost completely eliminated when the phase one training was improved.

CHAPTER 5

Temporal representation and the OSCAR model

5.1 Introduction

Previous models of serial order memory fail to take into account the *dynamic nature* of memory, relying instead on linked lists or item to node associations and forgetting by decay and interference (e.g. Ebbinghaus, 1913; Conrad, 1965; Shiffrin & Cook, 1978; Murdock, 1982). However, biological evidence is consistent with the idea that there exists a dynamic control signal which is, at least in part, responsible for a wide range of behaviour. This includes circadian based *time of occurrence* behaviour such as foraging (Beling, 1929, cited in Gallistel, 1990) and *temporal interval* behaviour such as the "trapline" behaviour of humming-birds (Gill, 1988, cited in Gallistel, 1990) that return to flowers at aperiodic intervals. More recently, Houghton (1990, 1994a) and Burgess and Hitch (1992, 1996; Burgess, 1995) have implemented models of serial order memory that use dynamic control signals during learning and recall. These models were reviewed in chapter 3; here we specifically address the nature of the control signal.

A dynamic control signal is required to possess a number of properties. Gallistel (1990) and Church and Broadbent (1990) suggest it should contain components from both fast and slower moving oscillators. In the following chapter, it is suggested that the signal should also satisfy two further requirements: the first, the *similarity* requirement that the control signal at any point should be highly similar to the signal at nearby points in time and much less similar to the control signal at a point much further away in time; and the second, a consequence of the first, is the *non-repetition* requirement, that the control signal must at no time repeat itself.

A number of attempts at modelling a temporal control signal exist. Church and Broadbent (1990), in an attempt at modelling interval estimation, implement a vector representation of the temporal control signal based upon Gallistel's (1990) insight into the representation of time. In Houghton's (1990) competitive queueing model for serial order, a temporal signal is implemented as a two-dimensional control signal comprising initiator and end node signals that decay and rise respectively. A third example is the context of Burgess and Hitch's (1992, 1996) model of serial recall, implemented as a large vector with a small non-zero component that travels across the width of the vector at each temporal step.

In the following chapter, these components and models of temporal signals are discussed. An oscillator based control signal, the *context vector* is introduced and its properties explored. Next, the context is coupled with a Hebbian associator to form the OSCillator-based Associative Recall (OSCAR) model of serial order memory. This model, in its most basic form, is shown to reproduce the serial position curve typical of serial ordered recall tasks (e.g. Baddeley, 1968).

5.2 Evidence for and properties required of biological clocks

There is evidence to suggest that a system of oscillators provide animals and plants with an internal representation of time vital for a wide range of behaviour. In animals, they are responsible for behaviours such as foraging for food (Beling, 1929; cited in Gallistel, 1990), estimating time and rate (Church & Broadbent, 1990; Gallistel, 1990), perception (Dehaene, 1993) and motor performance (Latour, 1967). Evidence suggests that similar mechanisms in plant life are responsible for opening leaves, releasing spores and flowering (Millar, Carré, Strayer, Chua & Kay, 1995).

Beling (1929, cited in Gallistel, 1990) observed that bees were able to record the time at which food was presented to them at specific locations and subsequently anticipate when food becomes available at that location. Studies demonstrate that representations in bee memory, such as odour, location and colour, are all linked to a temporal record of when the smell, location or sight was discovered (Gallistel, 1990).

Gill (1988, cited in Gallistel, 1990) observed that hummingbirds employed the use of a different timing mechanism to that of the bees and rats. Hummingbirds forage by "traplining" i.e. regularly visiting plants at specific locations. The issue for the hummingbird is when to visit a flowering plant. The longer it waits, the more nectar the plant will have produced. Conversely, the longer it waits, the more likely it is that a different hummingbird will have already visited that same plant. As these variables are aperiodic, so the hummingbird relies on being able to represent intervals between visits accurately, even to the extent that the ability to do so may constitute a selective advantage.

There is sufficient evidence to presume that animals (and plants) must have some temporal representation in order that they be able to measure temporal intervals, such as when the hummingbird decides on when next to visit a flower, and record time of occurrence, such as the bees and rats learn to anticipate feeding time. What is unclear is the precise nature of such timing mechanisms. Gallistel (1990) states that time could represented by the phase of a single oscillator. However, it would be impossible to distinguish intervals greater than the period of the oscillator.

This is analogous to using the second hand on a clock in order to represent intervals greater than a minute in duration. If at the start point, the second hand pointed to 17 seconds and at the end point, to 21 seconds, it would be impossible to know if the duration of the interval was four seconds or 64 seconds. Conversely, if the period were too long, it would be impossible to distinguish intervals much less than the period of the oscillator as the error in interpreting the interval may be similar in magnitude to the interval itself. Considering a clock face once more, this is analogous to using the hour hand in order to represent the interval. The hour hand has a period of 12 hours, so for a seven second interval, it would only rotate 0.06 degrees. This lack of resolution results in large errors when trying to estimate small intervals. Gallistel suggests that the solution is not to rely on the phase of a single oscillator, but to employ a number of oscillators and record the phases of each.

Although there are estimates of the frequency of human oscillators (Treisman, Faulkner, Naish & Brogan, 1990) there is speculation as to the range of frequencies required in order to account for such a wide range of behaviour (e.g. Church & Broadbent, 1990). Gallistel (1990) suggests that in a system of coupled oscillators, one could have a circadian period: a period that cycles approximately once a day. Others could have periods in the range of milliseconds, seconds, minutes, hours, weeks, months or years (Aschoff, 1981, cited in Church & Broadbent, 1990). By recording the time at which events occur and noting the phases of a series of oscillators, it would be possible for animals to compute intervals between events. Church and Broadbent (1990) suggest that if the fastest oscillator in a system had a period of two hundred milliseconds, and that each subsequent oscillator had a period of double the previous, only thirty oscillators would be necessary to ensure that the slowest oscillator would not cycle in the three and a half year lifetime of a rat.

This last point, that the slowest oscillator should not complete a cycle during the lifetime of the animal, leads to the first of two requirements of a dynamic context signal. That is that each context signal state should be unique and hence at no point during the systems lifetime should context states ever repeat themselves. As with the Church and Broadbent (1990) model, we can satisfy this, the *non-repetition requirement*, by ensuring that the frequency of the slowest oscillator in a system of coupled oscillators, is such that it will never complete a full revolution in the lifetime of the system. If the same method of period doubling as suggested by Church and Broadbent is employed, approximately thirty five oscillators would be sufficient for a human lifetime. The second requirement of the context signal is that neighbouring context signals, separated by a short temporal distance, must be more similar to each other than those separated by greater temporal distances. This property appears to be intuitive in that although we appreciate that moments close together in time are in fact distinct, they are less so than moments separated by much greater distances[8]. This is referred to as the *similarity requirement*. In order to satisfy this requirement with a system of coupled oscillators, there must be a distribution of oscillator frequencies

[8] Note that Burgess and Hitch (1992, p454) suggest the use of a context that has zero correlation between widely temporally separated states.

such that the effect of the fastest oscillators rotating between steps will have less of an impact upon the similarity of immediately neighbouring context states than the impact of the slower moving oscillators rotating on context states that are more widely separated. The consequence of the similarity requirement being satisfied is that the non-repetition requirement will also be satisfied.

Furthermore, we require the context signal to be *reinstatable*. The method by which each context vector is reinstated in sequence must be an integral part of the architecture. If this is not the case, then the problem of recalling a list of items becomes one of recalling the sequence of learning-context signals. Also, the mechanism underlying the context signal should be independently motivated and not constructed in a manner as to only address the memory phenomena it will be applied to.

Before outlining our proposed solution to the problem of modelling temporal context, a number of current solutions are described. Specific details of these models may be found in chapter 3, in the following section attention is drawn solely to the manner in which the temporal signal is implemented.

5.3 Models of the internal temporal signal

5.3.1 Church and Broadbent's storage vector

Church and Broadbent (1990) implement Gallistel's (1990) suggestion that temporal representation is possible given a number of coupled oscillators, each with one of a wide range of periods: from milliseconds to months and years in duration. For their model of interval timing, they choose to represent the current "time" by the state of an 11-element *storage* vector. Each element corresponding to one of 11 adjacent oscillators with periods in ratios of 2:1, the fastest of which rotates with a period of 200 milliseconds, the next with a period of 400 milliseconds, and so on until the 11th which rotates with a period of 204.8 seconds.

Church and Broadbent suggest that although animals may not be able to resolve the phase of an oscillator with a high degree of accuracy, they may instead be able to recognise the *half-phase* i.e. + or - within each period. As such, the vector element representing the phase of the oscillator with which it is associated appears not to change continuously with the phase of the oscillator, instead it appears to toggle back and forth between two values: in the present case, +1 or -1. The net result is a vector that appears very similar in behaviour to a binary counter as is evident from figure 5.1.

t_1	- + - + - - + - + + -
t_2	- + - + - - + - + + +
t_3	- + - + - - + + - - -
t_4	- + - + - - + + - - +
t_5	- + - + - - + + - + -

Figure 5.1 *Five successive states of the 11-element Church and Broadbent storage vector*

By representing time in this manner, Church and Broadbent's signal could satisfy the non-repetition property of a context signal only if the period of the slowest oscillator was sufficiently long. In the present case, this is only 204.8 seconds, deemed too short to be a realistic value. However, by increasing the capacity of their storage vector, they could accommodate much slower moving oscillators and hence satisfy the non-repetition property within a realistic animal lifespan.

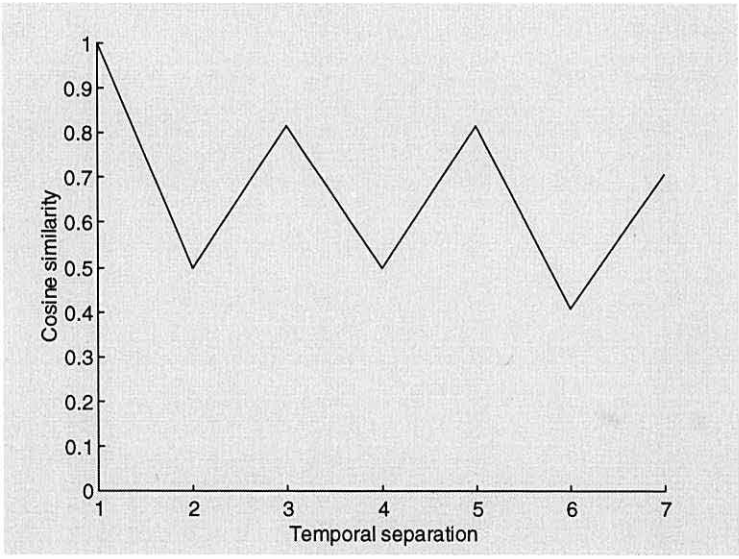

Figure 5.2 *Cosine similarity curve for the Church and Broadbent storage vector*

However, the Church and Broadbent signal does not satisfy the similarity property. Examining the cosine between the start signal (eleven zeroes and a single one representing *food present*) and seven subsequent steps, it is clear that although the overall trend is for the signal to become less like the initial signal, there are many occasions where confusions between *equally similar* states could occur. For example, the storage vectors three and five steps after the start signal each have an identical cosine of approximately 0.8 (figure 5.2).

5.3.2 Houghton's control signal

Houghton (1990) describes a competitive queuing model of serial order. Fundamental to the performance of the model is the behaviour of the *sequence node* layer of the network, which contains connected pairs of initiator (or start) and end nodes: I-nodes and E-nodes. These are used to represent the "temporal edges" that surround a temporally ordered pattern such as a word.

During learning, at the onset of a word, an uncommitted I-node, which is initially fully activated, decays with each successive time step until the word is complete. At this point the corresponding E-node becomes fully active. In this manner, every element of the word will correspond to a particular state of the appropriate I and E-node pair.

The behaviour of this two dimensional representation of context is explored further in a later report (Houghton, 1994b). Houghton reaffirms that the control signal should possess two components: the first a rising (i.e. the E-node) and the second falling (i.e. the I-node).

Figure 5.3 illustrates seven vectors (i.e. one for each of the learned items) created as a result of the function of the combined E-node and I-node activations. From this is it is evident that, if the angle between vectors is used as the measure of similarity between successive states, the control vectors at either end of the list are most likely to be confused. This is confirmed by examining the cosine similarity curve generated by

plotting the cosine between the first control vector and each of the subsequent vectors illustrated in the previous figure.

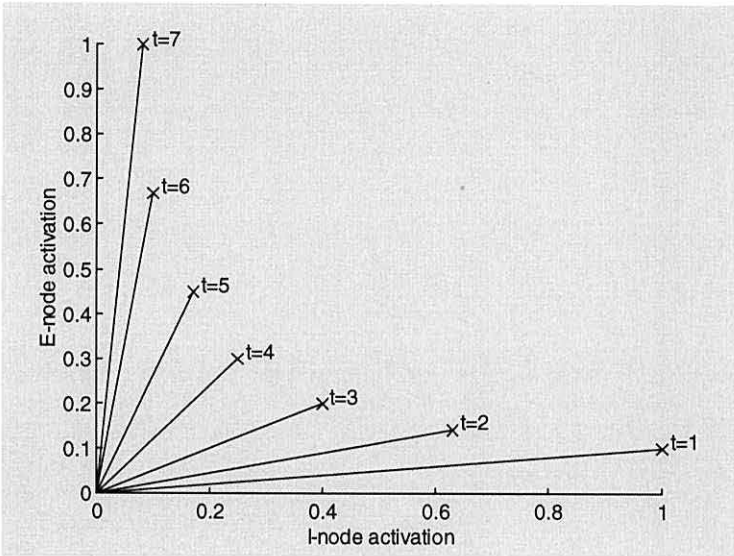

Figure 5.3 *I-node and E-node activations for a seven item sequence (Adapted from Houghton, 1994b, figure 1)*

Clearly, the control signals immediately adjacent to the reference signal are very similar to that reference. However, the control signals become different very quickly in the central positions (i.e. t=3, t=4 and t=5) where the cosine between them and the reference signal decreases rapidly.

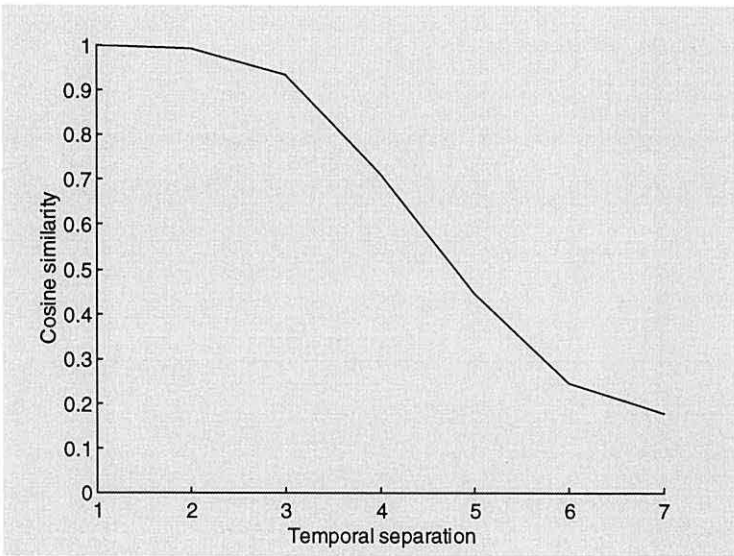

Figure 5.4 *Cosine similarity curve for the Houghton I-E node control signal*

It is clear from figure 5.4 that the Houghton control signal satisfies the similarity property required of an effective context: neighbouring control signals are more similar to each other than those separated by a wider temporal distance. However, the similarity property exhibited by the control signal is linked closely to its ability to satisfy the second, non-repetition property. Given such a limited dimensionality control signal, if a number of control vectors were required, although controls would not need to be repeated (and as such the control signal satisfies the non-repetition property) the increased number of vectors within the single two dimensional quadrant of space available to the control signal could result in a set of vectors that are very similar to each other. Therefore, the similarity property of the signal may be affected by the change in inter-control vector spacing.

5.3.3 Burgess and Hitch's "context"

Burgess and Hitch (1992), in an implementation of Baddeley's (1986) articulatory loop model of short-term memory, develop a representation of the nonphonological and temporal information presented to the model, the *context*. They implement this as a random pattern of activation that alters progressively with time. However, because the context is generated randomly, it is unclear how it may be reinstated reliably during recall.

The context vector has a dimensionality of 50 elements (nodes) and when each new item is presented to the model, a random two-thirds of these nodes are updated. All but six of these updated nodes are reset to zero, the remainder are given a nonzero activation that reflects the average activation in the model's phoneme layer. On average, nine of the context nodes are active for each new item.

By ensuring that some nodes are unchanged when new items are presented, there is some overlap of contexts and hence temporal correlation between contexts for successive items. Burgess and Hitch note that when using a context vector of this form, the similarity of activation between neighbouring items will vary monotonically with their separation.

However, in assessing the properties of the model using such a context signal, Burgess and Hitch suggest a modification of the basic context vector in order that transposition errors be localised to either side of the current item. This is achieved by ensuring that there is zero correlation between temporally well separated states (table 5.1).

Table 5.1

Three successive states of the modified Burgess and Hitch context signal (Adapted from Burgess & Hitch, 1992)

Time	Activation of context nodes										
t_1	*	*	*	*	*	*	0	0	0	0	0
t_2	0	0	*	*	*	*	*	*	0	0	0
t_3	0	0	0	0	*	*	*	*	*	*	0

Examination of the cosine similarity function for this modified Burgess and Hitch context (figure 5.5) confirms that the similarity between a fixed context and subsequent contexts decreases linearly as the temporal separation increases. The cosine between the reference context and that one step later is 0.66, with the context two steps later is 0.33, and zero with all other contexts.

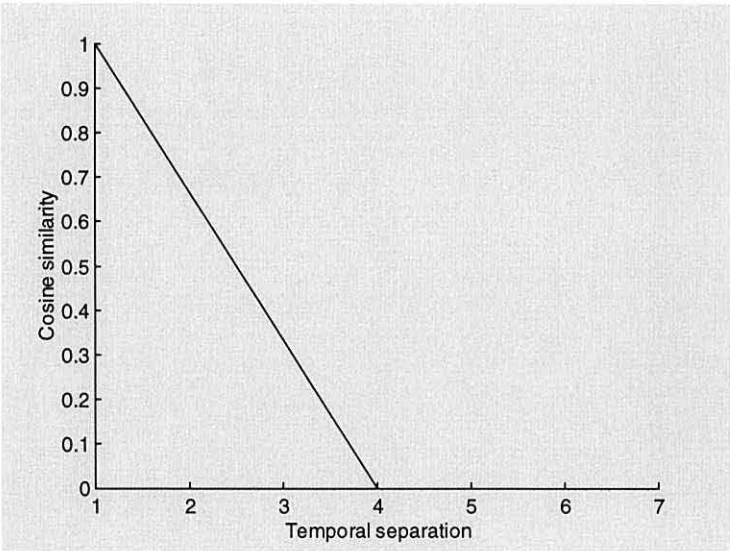

Figure 5.5 *Cosine similarity curve for the Burgess and Hitch context vector*

It is clear that the later Burgess and Hitch context signal satisfies the similarity and reinstatability properties required of a temporal signal. The similarity is greater for closely spaced contexts than it is for those more widely separated. However, it is difficult to know whether, over an arbitrary period of time, the Burgess and Hitch

context vector could be extended to satisfy the non-repetition property. If the context vector is updated in the manner suggested, the number of separate context vectors that could be modelled is limited by the dimensionality of the context vector. It is unclear what the active elements of the context signal should do once they have "travelled" the full width of the context. However, the impression is that the Burgess and Hitch context vector does *not* satisfy the non-repetition requirement for a context signal, as the only option available to them when the pattern of activation reaches the end of the context is to return to the opposite end of the context.

5.4 An oscillator based control signal: the context vector

5.4.1 Introduction

In the following section, a temporal control signal composed of a number of coupled oscillators with a wide range of frequencies, the *context vector*, is introduced. Not only must the context vector satisfy the two properties described previously: the non-repetition requirement and the similarity requirement, but it should also remain normalised for each discrete step in order to improve its applicability to learning and recall modelling tasks. Having described the generation of such a signal, its properties are explored in detail prior to its application to the problem of modelling serial ordered memory.

5.4.2 The context vector

Developing the ideas of Gallistel (1990) and Church and Broadbent (1990), it is possible to generate a model of the temporal context by combining a number of oscillators with a wide range of frequencies in order that different elements of the context will evolve at different rates. The rate at which the context will evolve will depend upon the frequency of each of the oscillators and the manner in which they are combined to produce each context element. By careful consideration of the distribution of the oscillators to each of the context elements, it is possible to generate a context vector that satisfies the non-repetition and similarity requirements.

By assuming that the fastest moving oscillators are common to only a few of the context vector elements, it is possible to satisfy the similarity property. This can be explained by considering two adjacent contexts (i.e. little temporal separation). Comparing the state of the oscillators at both times, it is apparent that the slowest evolving oscillators will have changed very little, if at all. Even with the fastest evolving oscillators, they too will have evolved very little in the time between the two contexts. However, as the temporal distance separating the two contexts increases, so the difference between the states of the fastest moving oscillators will become more apparent and hence the context vectors will become increasingly dissimilar as more of the oscillators evolve away from their state at the time of the first context.

Furthermore, by ensuring that elements of the slowest moving oscillators are common to each of the context vector elements, it is possible to satisfy the non-repetition property of the dynamic signal. Simply, if we assume that the slowest oscillator will never complete a cycle in the system's lifetime, so we can ensure that the state of the context will never be identical to any other. However, if there are insufficient slow moving oscillators common to each of the elements of the context signal, then although the context could still satisfy the non-repetition property, it is very likely that widely separated contexts could become very similar to each other. To avoid this, we must ensure that sufficient slow moving oscillators are common to each of the context elements.

We also suggest that if the context is developed using a system of predictable oscillators, it should be possible to reinstate a sequence of context states by reinstating just the first of the sequence.

In summary, the fast evolving components of the context will serve to distinguish between contexts separated by short temporal distances (and hence satisfy the similarity requirement), whilst the slow moving components of the context will serve to distinguish between contexts separated by large temporal distances (and satisfy the non-repetition requirement). In order to ensure that the context satisfies both of these

requirements and in order to maintain normalisation[9], oscillators are coupled together in the manner illustrated by figure 5.6.

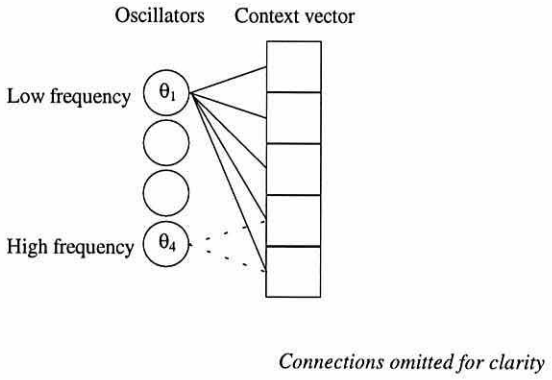

Figure 5.6 *Context vector generation from oscillator array*

Although the connections from the central pair of oscillators have been omitted in order to improve clarity, this figure illustrates how the low frequency oscillator is connected to each context vector element, while the high frequency oscillator is linked to only a subset of the oscillators. More precisely, the slowest oscillator (θ_1) is common to each of the context vector elements whilst the fastest evolving (θ_4) is common to only two elements. In this manner, the slowest moving oscillators contribute most to the overall state of the context while the fastest moving oscillators contribute the least.

$$
\begin{array}{lclcclcl}
\mathbf{c}(1) & = & \cos(\theta_1) & * & \cos(\theta_2) & * & \cos(\theta_4) \\
\mathbf{c}(2) & = & \cos(\theta_1) & * & \cos(\theta_2) & * & \sin(\theta_4) \\
\mathbf{c}(3) & = & \cos(\theta_1) & * & \sin(\theta_2) & * & \cos(\theta_5) \\
\mathbf{c}(4) & = & \cos(\theta_1) & * & \sin(\theta_2) & * & \sin(\theta_5) \\
\mathbf{c}(5) & = & \sin(\theta_1) & * & \cos(\theta_3) & * & \cos(\theta_6) \\
\mathbf{c}(6) & = & \sin(\theta_1) & * & \cos(\theta_3) & * & \sin(\theta_6) \\
\mathbf{c}(7) & = & \sin(\theta_1) & * & \sin(\theta_3) & * & \cos(\theta_7) \\
\mathbf{c}(8) & = & \sin(\theta_1) & * & \sin(\theta_3) & * & \sin(\theta_7)
\end{array}
$$

Figure 5.7 *Composition of the 8 element context vector, **c***

In order to simplify the computational process involved in modelling the contexts, and in order to improve the recall process, we ensure that each context vector is normalised. Hence, the solution we suggest here is an n-dimensionality context vector,

[9] To facilitate item recognition and recall.

c, generated by combining the sine and cosines of n-1 oscillators. The context vector, **c**, illustrated in figure 5.7, contains eight elements, **c**(1) to **c**(8). Each is the product of three sine or cosine terms. These terms take the angle (in radians) of each of the oscillators, θ_m as their input. In contrast to the models of Church and Broadbent (1990), Houghton (1990) and Burgess and Hitch (1992), these are initialised with random angles: i.e. the initial state of the context is arbitrary. However, each of θ_m is incremented by a small amount, $\delta\theta_m$ radians[10] after each simulated time cycle. The magnitude of $\delta\theta_m$ increases in some relation with m. Therefore, as θ_1 has the smallest increment and hence the slowest frequency, it is common to each of the context vector elements. Conversely, θ_3, with a higher frequency, is common to only half of the elements. However, as θ_7 takes the largest value of $\delta\theta_m$, and hence evolves with the highest frequency, it is common to only the last two elements of the vector.

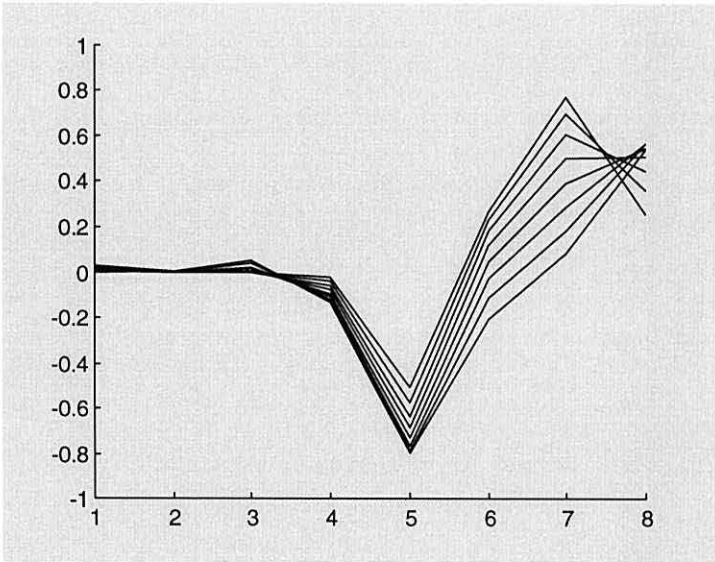

Figure 5.8 *Eight bit context vector evolving through eight successive states*

By linking elements of the context vector to the oscillators in such a manner, it is possible to cluster the fastest and slowest evolving components at opposite ends of the context vector (e.g. figure 5.8). This has the advantage that is will be possible to vary the nature of the reinstated context vectors used during recall. For example, one might

[10] Note that all angles are recorded in radians. There are $2*\pi$ radians in a complete revolution and therefore $\pi/4$ radians is equal to 45°. Where $\delta\theta$ values are given it may be assumed that the values are given in radians unless stated otherwise.

wish to copy the fastest evolving elements of the last context vector to all of those used during learning in order to examine the effects of immediate recall.

However, although it has been stated that the oscillators should vary in frequency, and that the range of frequencies should be widely distributed in order that the signal should satisfy the properties outlined previously, the precise nature of these frequencies has not yet been described.

In the following section, the effect of implementing this template for a context vector with different ranges of $\delta\theta_m$ is examined.

5.4.3 Exploring the properties of the context vector

In the previous section, the structure of the context vector was outlined: a system of coupled oscillators with different frequencies, combined in such a manner as to ensure that the fastest and slowest evolving components may be localised and that the vector remains normalised for each step. In the following section, two different methods for distributing the range of oscillator frequencies are considered. Further investigations are presented in the appendix.

5.4.3.1 Simulation 8: *Constant delta theta size*

Introduction

In this first simulation, each of the oscillators, θ_m, is incremented at each discrete step by the same amount, $\delta\theta_m$.

Method

A sixteen element context vector was constructed in the manner as described in section 5.4.2. Each θ_m was seeded with a random angle such that the first context vector was at some arbitrary state. The values of $\delta\theta_m$ were calculated, in this case they were constant for all m. Each θ_m was incremented by the constant (equal to $\pi/16$) and the next context vector was generated. 32 successive states of the context were

calculated and the similarity, measured as the cosine, between every context with every other was recorded and averaged over 100 different trials in order to minimise any effect due to the initial state of the context.

Results

It is clear from figure 5.9 that the context vector generated in this manner fails to satisfy the non-repetition property as after approximately 8 steps the contexts start to become increasingly similar to the original context. Finally, after 16 steps, the context is identical to the original context.

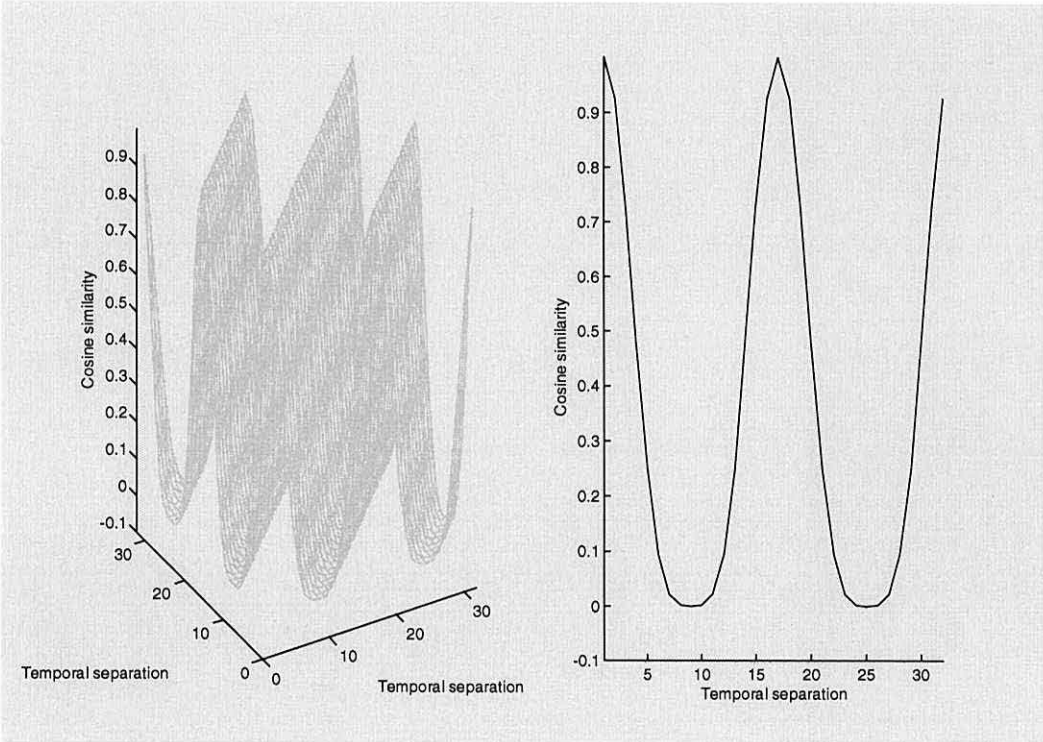

Figure 5.9 *End-on and cut-away view of cosine between neighbouring context vectors*

Discussion

Analysis reveals that the results of simulation eight are exactly as would be expected. If the inner product between the first context and a second at some arbitrary time t is reduced, it reveals that the inner product (or similarity cosine as the contexts are normalised) equates to:

$$\mathbf{c}_0.\mathbf{c}_t = \cos^4(\delta\theta\ t) \qquad (5.1)$$

i.e. the inner product between neighbouring eight bit contexts, when $\delta\theta_m$ is constant for all m, is equal to the cosine to the fourth function of $\delta\theta$ at time *t*. It also illustrates that the initial state (i.e. position) of the context has no bearing on its behaviour during subsequent states. It is influenced solely by the small increase in θ.

Subsequent analysis has confirmed that for a 2^n element context vector, if $\delta\theta_m$ is constant for all m, then the inner product may be expressed as:

$$\mathbf{c}_0.\mathbf{c}_t = \cos^n(\delta\theta\ t) \qquad (5.2)$$

However, as these contexts clearly fail to satisfy the non-repetition requirement of a temporal signal, they must be rejected in favour of a more refined context.

5.4.3.2 Simulation 9: *Non-linear delta theta size*

Introduction

A number of different methods for generating and combining the oscillators have been considered including the use of a linear relationship between one oscillator phase and the next. However, we suggest that, given the shortcomings of these methods and in line with previous independently motivated models that consider the requirements of natural oscillators (e.g. Church and Broadbent, 1990), a wide distribution of frequencies such as that provided by a 2^m (where *m* corresponds to the mth context vector element) function be employed. Therefore, the highest frequency (and hence shortest period of oscillation) could represent milliseconds whilst the lowest frequency (e.g. with a periodicity of $0.001*2^{16}$ seconds) would be more than adequate to ensure that there could be no repetition during the lifetime of the system.

Method

An eight element context vector was constructed in the same manner as described previously and illustrated in figure 5.7. Each of θ_m was seeded with a random angle such that the first context vector was at some arbitrary state. However, the increment in θ_m at each discrete step, $\delta\theta_m$ was increased non-linearly by scaling with 2^m:

$$\delta\theta_m = rand*0.02*2^m \quad (5.2)$$

Note that *rand* is a small random value uniformly distributed between zero and one, that helps to produce a smoother performance. In order to prevent the possibility of the random scaling value having sufficient small magnitude to effectively transpose the $\delta\theta_m$ values (e.g. $0.99*0.02*2^2 > 0.001*0.02*2^3$) each of the $\delta\theta_m$ values is sorted into ascending order. By averaging over sufficient trials it is presumed that the 2^i gradation will hold.

Each of θ_m was incremented by the respective value and the next context vector generated. 32 successive states of the context were calculated and the similarity between each averaged over 100 different trials.

Results

Figure 5.10 confirms that the measure of similarity between the reference context vector and those before and after that vector decreases as the temporal spacing between them increases, in either direction, from the reference. Furthermore, the similarity between the reference vector and those towards the extremes of the system lifespan remains approximately, or below, zero.

Discussion

This method for generating context vectors is deemed the most reliable option. The similarity requirement appears to be satisfied as the contexts separated by the smaller temporal distances are more similar to the reference context than those separated by greater temporal distances. However, it is clear that the similarity function drops steeply in an almost concave manner from the start point. This may be deemed inappropriate as we anticipate that, in order for the signal to satisfy the similarity and non-repetition properties, those contexts immediately neighbouring the reference should be more similar (cf. Houghton's I-E node *context* of figure 5.4).

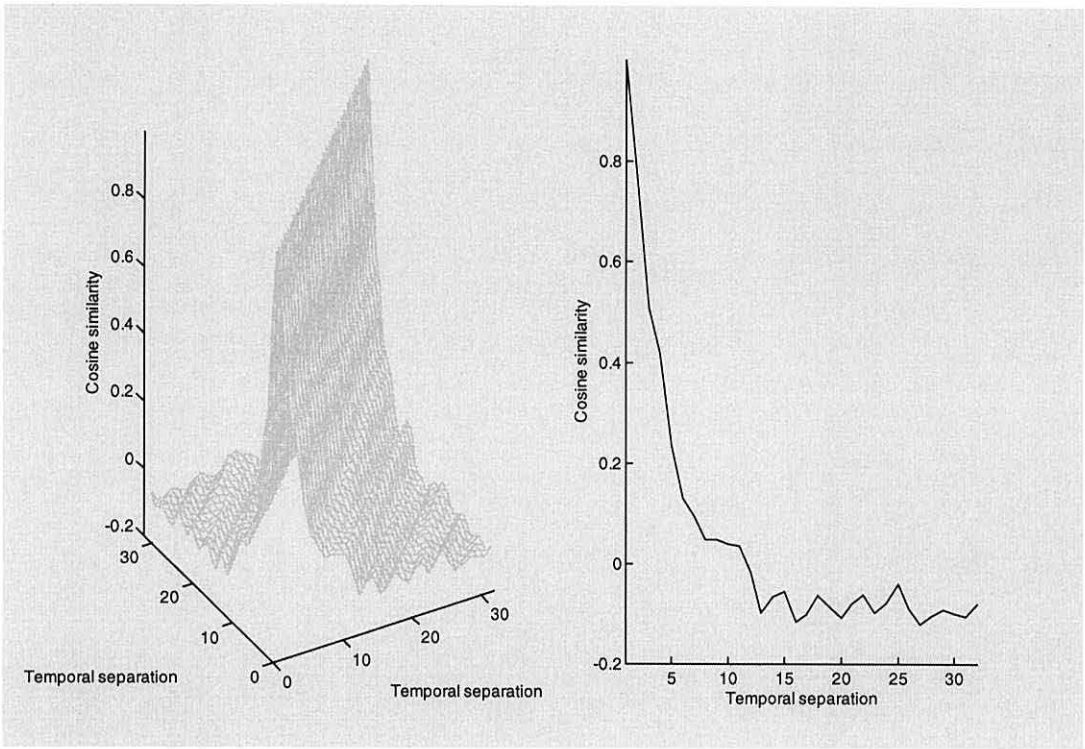

Figure 5.10 *End-on and cut-away view of cosine between neighbouring context vectors*

We can be sure that this context will satisfy the non-repetition property as the slowest evolving oscillator will have sufficiently slow frequency (e.g. approximately 0.04 radians) for this system.

5.4.4 Summary

In summary, we have shown that it is possible to generate a plausible representation of the dynamic internal state, or context, during learning, using a system of oscillators with a wide range of frequencies. These have been shown to satisfy both of the requirements we suggest for a context signal. Firstly, the similarity property, which states that the similarity (measured by the inner product or cosine between a pair of context signals) should decrease with increasing temporal separation. Secondly, the non-repetition property which states that the context vectors should not return to any previous state during the lifetime of the system. The present oscillator based context signal is of a much greater dimensionality to that of Houghton's (1990; 1994) and it is anticipated that this improved capacity will mean that fewer inhibitory mechanisms will be required in order to produce serial ordered behaviour.

We now wish to apply the idea of a dynamic context signal to the problem of memory for serial order. In the next section we will outline one solution as to how this problem may be solved.

5.5 An oscillator based associative recall model of memory: OSCAR

5.5.1 Introduction

In this section, the context vector described in section 5.4 is coupled with a simple associator to model the storage and recall of sequences of items. Items are represented by normalised vectors containing elements taken randomly from a set of scalars, normally distributed about zero with variance of one. Each context corresponds to a discrete moment in time which in turn corresponds to the precise moment when the appropriate item was presented for learning during training. Learning of the item-to-context associations is by simple Hebbian association with the additions that the memory trace, or weights, decay over time and that items may be stored in the memory trace with decreasing strength as their displacement from the first item increases.

Table 5.2

Summary of the learning and recall sequence employed by the context based OSCAR model

Step	Learning process
1	Generate context vector
2	Associate stimulus item with context vector by Hebbian learning
3	Decay current contents of memory trace
4	Store latest association in the memory trace
5	Evolve the context vector by incrementing each oscillator
6	Return to *step 1* until every stimulus item has been learned

Step	Recall process
1	Reinstate learned context vector
2	Probe memory trace with context vector
3	Compare retrieved item with lexicon of recallable items
4	Recall lexicon item most similar to retrieved item
5	Update lexicon of recallable items
6	Return to *step 1* until every stimulus item has been recalled

Recall involves reinstating the dynamic context signal and presenting it as a cue for recall in order that an approximation to the item associated with that context during learning be generated as the output. This approximation is then compared with a vocabulary containing both items presented during learning and a number of previously unseen distractor items. The item it most closely resembles, measured by taking the cosine between the two vectors, is recognised as the item retrieved from memory. Recall of subsequent items is by reinstating each of the corresponding contexts and presenting them as cues to the system. Typically a simple inhibitory mechanism prevents items being recalled repeatedly. This process is summarised in table 5.2.

5.5.2 Modelling sequential events with the context vector

In order to simplify the modelling of time for sequence memory, one further assumption must be made. Time should be considered to be not continuous, but instead a sequence of discrete temporal steps. These steps may in fact be so small as to give the impression of a continuous sequence. However, to each of these discrete moments in time we assign a unique context vector representing the internal cognitive state of the system at that moment in time (cf. *time tags*; Estes, 1972). Therefore, at any moment in time, t_n, there is a corresponding context, $\mathbf{c}_n$, that represents the state of the system at time t_n. This may be expressed more precisely as:

$$t_n \leftrightarrow \mathbf{c}_n \tag{5.3}$$

Therefore, if items are presented at discrete intervals during learning, these items will be associated with the context that corresponds to the particular moment in time when the item was presented.

t_{-2}	-	$\mathbf{c}_{-2}$	-	$\mathbf{i}_1$
t_{-1}	-	$\mathbf{c}_{-1}$		
t_0	-	$\mathbf{c}_0$	-	$\mathbf{i}_2$
t_1	-	$\mathbf{c}_1$		
t_2	-	$\mathbf{c}_2$	-	$\mathbf{i}_3$

Figure 5.11 *Associating items presented at discrete intervals to the contexts that correspond to those moments in time*

This is best illustrated by figure 5.11 where the latest item presented is i_2 and hence all the time steps and contexts are numbered relative to that position. Figure 5.12 illustrates the effect of using six "highly distinct" contexts for the item association. The similarity value is unity for each context in its target serial position. However, it is clear that, taking the context vector in the third serial position as an example, the context from the previous serial position and that for the next serial position both have similarities of approximately 0.4 in the third serial position. More specifically, the third context is highly unlikely to be confused with those contexts in the second and fourth positions because in the present serial position, they report very low measures of similarity. This can be seen to confirm the intuitive belief that a set of "highly distinct" contexts will be easily distinguished from each other.

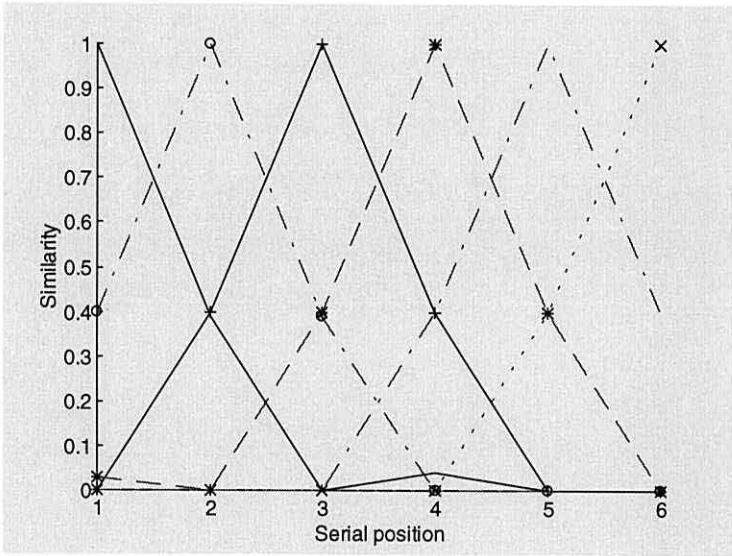

Figure 5.12 *Similarity of six "highly distinct" contexts*

Figure 5.13 illustrates the similarity of a set of "less distinct" contexts. Once again taking the third serial position as a reference, that the similarity measure for the second and fourth contexts at the third serial position, is much higher than previously (a cosine of approximately 0.85). Furthermore, whereas before the similarity measure was effectively zero beyond the serial position immediately neighbouring the reference position, in this case the similarity measure for the first and fifth contexts at the third serial position is in fact around 0.6. Again this confirms the intuition that if the context

vectors are evolving more slowly, so neighbouring contexts are going to have more elements in common, i.e. more overlap, and hence be more similar to each other.

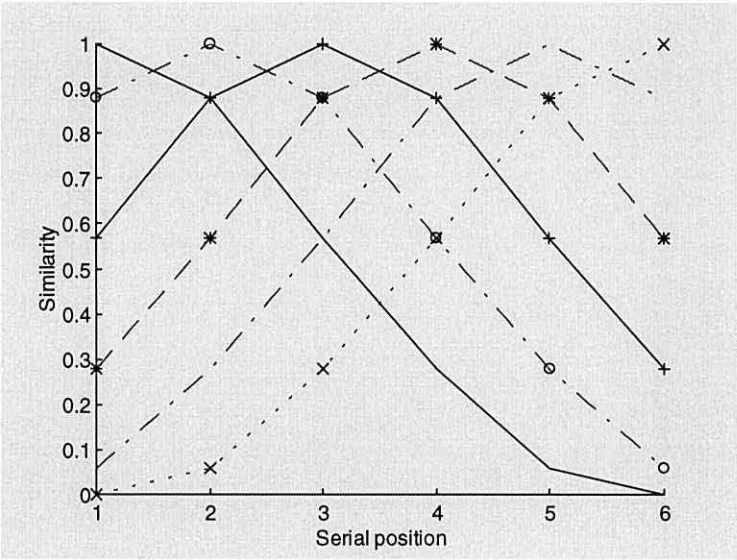

Figure 5.13 *Similarity of six "less distinct" contexts*

However, it is apparent that if the "less distinct" contexts were sampled at greater intervals, they would appear to behave like "highly distinct" contexts. This is a useful observation as it suggests that one set of context vectors is sufficient and that by manipulating the spacing between to-be-associated contexts, the distinctiveness between them may be controlled. In this manner the degree of confusion at cueing may be limited.

We also suggest that this property will give rise naturally to order errors. If the "less distinct" contexts are used during learning, and the third context reinstated at recall and presented as a retrieval cue, it is clear that it is highly likely to become confused with the retrieval cue for both the second and the fourth serial positions. Therefore, there is a good possibility that confusion will occur resulting in either the second or fourth item being recalled erroneously in the third serial position. Furthermore, we predict that as the retrieval cues for the first and last serial position can only be confused with the contexts immediately following *or* preceding them, so there is less probability of an error occurring at these end positions. The consequence of this is that

more errors should occur in the central serial positions resulting in a bowed serial position curve.

However, it was stated during section 5.4.3 that in order to produce the similarity function from the set of contexts, the similarity function had to be averaged for a number of different sets of contexts. This appears to have serious implications for learning if one novel set of contexts is insufficient to ensure that the model will behave as predicted. However, we argue that this processes of averaging over a number of sets of contexts is akin to the use of one larger, high dimensionality "overall context" that consists of each of these smaller, low dimensionality "contexts". Simply, this allows for a smoother and more predictable performance, and therefore for the remainder of this thesis, where averaging the contexts is discussed, it should be thought of as simply resulting in a higher dimensionality context vector.

A further issue to be considered when applying the context signal to the problem of sequence memory is how to update the context signal before recall. This problem can be approached in either of two ways: the first, and simplest, method is to reinstate the context exactly as it was during the learning stage. However, the second, and preferred approach, is to partially reinstate the context. We assume here that recall begins almost immediately after the last item is presented. We implement this by copying the fastest moving elements of the context-at-recall to all of the contexts prior to recall. We expect that this could preferentially influence the most recent items and hence reduce primacy in the serial position curve.

In fact, the majority of the data presented later in this section is generated by the former method although some investigation of the latter is described.

In summary, we suggest that there exists a unique representation of internal state, a context, and that these contexts may be assigned to discrete moments in time. Further, by associating items, presented to the system at regular intervals, to the corresponding contexts, it is possible to model serial ordered behaviour. We have also demonstrated how the rate of context evolution and inter-context spacing are analogous to each

other and have suggested how they may influence performance. In the next section, the question of how the model can store these associations by simple Hebbian learning is addressed.

5.5.3 Learning by Hebbian association

We have already outlined how serial order information may be modelled by associating items presented sequentially with the appropriate context state. Now we examine how this may be simulated using a Hebbian associator network and single trial learning.

Let each item and context be represented by the p-by-1 dimensionality vectors **i** and **c** respectively:

$$\mathbf{i}_k = \begin{bmatrix} i_{k1} \\ i_{k2} \\ \vdots \\ i_{kp} \end{bmatrix} \quad \mathbf{c}_k = \begin{bmatrix} c_{k1} \\ c_{k2} \\ \vdots \\ c_{kp} \end{bmatrix} \tag{5.4}$$

Each item vector element is a scalar selected at random from a normalised distribution about zero with a variance of one. In order to facilitate the retrieval process, each item vector is also normalised. Items are presented sequentially to the network. As each item is presented, so it is mapped to a unique context vector:

$$\mathbf{c}_k \rightarrow \mathbf{i}_k \qquad \text{k=1, 2 ... q} \tag{5.5}$$

Here q corresponds to the number of item-context associations to be stored by the network during the learning phase, or list-length. It is possible to refine equation 5.5 with the introduction of **W**(k), the weight matrix generated as a result of the single item-context association, k:

$$\mathbf{i}_k = \mathbf{W}(k)\mathbf{c}_k \tag{5.6}$$

Therefore, the association between the item, $\mathbf{i}_k$, and the context, $\mathbf{c}_k$, may be stored in the weight matrix $\mathbf{W}(k)$. However, at present this only accounts for the association of a single item-context pair of vectors. As each item is presented sequentially to the network, so each association generated dynamically must be stored in the memory, in this case the associative memory matrix $\mathbf{M}$:

$$\mathbf{M} = \sum_{k=1}^{q} \mathbf{W}(k) \tag{5.7}$$

Hence equation 5.7 describes the accumulation of each association in order to produce the memory trace $\mathbf{M}$. It is also possible to express the storage procedure in terms of the contents of the memory matrix as a function of the list position:

$$\mathbf{M}_k = \mathbf{M}_{k-1} + \mathbf{W}(k) \qquad k=1, 2 \ldots q \tag{5.8}$$

Equation 5.8 illustrates that, for the kth item-context pair, the contents of the memory matrix $\mathbf{M}_k$ are equal to the contents of the memory trace as a result of the previous association, $\mathbf{M}_{k-1}$ plus the weight matrix generated by the association of the kth item-context pair.

Furthermore, we can express an approximation to $\mathbf{M}$ in terms of the stimuli vectors as:

$$\hat{\mathbf{M}} = \sum_{k=1}^{q} \mathbf{i}_k \mathbf{c}_k^T \tag{5.9}$$

Here, $\hat{M}$ represents an approximation to the contents of the memory matrix, $\mathbf{M}$, after each item-context vector pair has been presented. The vector $\mathbf{c}_k^T$ represents the transpose of the context vector $\mathbf{c}_k$ and q once again represents the total number of items in the list. Note that the term $\mathbf{i}_k\ \mathbf{c}_k^T$ represents the outer product between the probe pattern, $\mathbf{c}_k$, and the memorised pattern, $\mathbf{i}_k$. As both of the stimuli have matrix dimensions of p-by-1 so the outer product will have square dimensions p-by-p. Hence we can further refine equation 5.8 to incorporate equation 5.9:

$$\hat{\mathbf{M}}_k = \hat{\mathbf{M}}_{k-1} + \mathbf{i}_k \mathbf{c}_k^T \tag{5.10}$$

Therefore, the Hebbian learning process may be expressed in vector notation as described by equation 5.10. The memory matrix $\mathbf{M}_k$ is equal to the contents of the memory trace as a result of the previous association, $\mathbf{M}_{k-1}$ plus the outer product between the key pattern, the context vector $\mathbf{c}_k$, and the memorised pattern, the item $\mathbf{i}_k$.

The storage process is clarified by the signal flow diagram of figure 5.14. This illustrates how the nth item and context vector pair are associated together and combined with the contents of the memory trace, $\mathbf{M}_{n-1}$. Next the updated contents of memory, $\mathbf{M}_n$ are delayed whilst the next item and context vector pair are presented to the network.

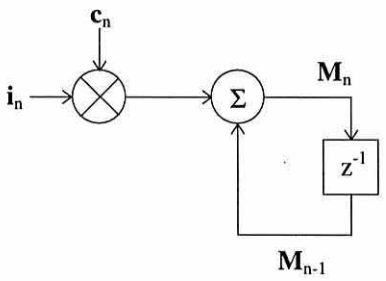

Figure 5.14 *Open loop implementation of OSCAR*

We have described the process of item, context, association and storage by Hebbian learning in a memory matrix. Next we address the problem of item recall and recognition.

5.5.4 Recall and Recognition

During item recall, the context vectors are presented sequentially to the memory trace in the same order as they were originally stored during the learning stage. Refining equation 5.6 to include the approximation to $\mathbf{M}$, $\hat{\mathbf{M}}$, and considering the output generated, $\mathbf{i}'$, when the context vector, $\mathbf{c}_j$, is presented to the memory matrix $\hat{\mathbf{M}}$:

$$\mathbf{i}' = \hat{\mathbf{M}}\mathbf{c}_j \tag{5.11}$$

Substituting equation 5.9 and expanding this as follows:

$$\mathbf{i}' = \sum_{k=1}^{q} \mathbf{i}_k \mathbf{c}_k^T \mathbf{c}_j \tag{5.12}$$

It should be apparent from equation 5.12 that $\mathbf{c}_k^T\mathbf{c}_j$ is the scalar inner product between the context probe and the context stored in the original association, $\mathbf{c}_k$. Therefore it is possible to simplify equation 5.12 as we can assume that all the context vectors are normalised, to:

$$\mathbf{i}' = \left(\mathbf{c}_j^T \mathbf{c}_j\right) \mathbf{i}_j + \sum_{\substack{k=1 \\ k \neq j}}^{q} \left(\mathbf{c}_k^T \mathbf{c}_j\right) \mathbf{i}_k \tag{5.13}$$

It is evident from equation 5.13 that if the context, $\mathbf{c}_j$ is reinstated and used as the probe presented to the memory trace, the retrieved vector will contain the target item, $\mathbf{i}_j$ and some noise. The strength of the noise will depend upon the similarity between the probe and every other context vector.

However, the item recall process is not yet complete. A further stage of item recognition is required in order for the correct item to be reinstated at the output. The retrieved item, **i'**, is presented to a lexicon of recallable items containing uncorrupted copies of each item presented to the model during learning along with a number of distractor items (the number of distractor items is typically either zero or equal to the number of items presented). As the recalled item is presented to each vocabulary item, so the cosine, once again the measure of similarity, between the recall item and each of the vocabulary items is calculated. The vocabulary item that has the greatest similarity (i.e. the highest cosine value) to the recalled item is recalled from the vocabulary and presented as the output.

We have outlined how it is possible to recover an item from the OSCAR memory matrix given the reinstatement and subsequent presentation of a context vector to the network. Furthermore, we have shown that the quality of the recalled item approximation depends upon the similarity (i.e. distinctiveness) of the context vectors. We have also explained how the recalled item is compared with a vocabulary of items, possibly including a number of distractor items, and how the item to which recalled item bears most similarity, is retrieved from this vocabulary and presented as the models output.

In the following section we demonstrate how the model described so far is capable of performing serial ordered learning and recall, and further that it is already capable of reproducing the distribution of order errors observed by Estes (1972).

5.6 Simulation 10: *Serial ordered learning and recall*

Introduction

In order to demonstrate that the ability to perform serial ordered learning and recall occurs as a natural consequence of the OSCAR architecture, a simple network implementing the storage and recall rules described thus far was applied to the problem of learning and recalling a sequence of six items. We anticipate that the model should be capable of producing a serial position curve that includes a degree of primacy (improved performance for the early items) and recency (an improvement in the last serial position). Examination of the transposition curve should confirm that order errors occur most often in the serial positions immediately adjacent to the target serial position.

Method

The following method will be employed, with only minor alterations to parameter values, for each of the OSCAR simulations. Therefore, here it is described in detail, while in subsequent sections we will refer the reader back to this section, highlighting any methodological differences.

OSCAR was presented with previously unseen lists of six items. Items were normalised and of dimensionality 16. Vector elements were drawn randomly from a normal distribution about zero with variance of one. During training, each item was associated with a 16-element context vector by Hebbian association and the association accumulated in the composite memory store. Recall involved reinstating each context vector from the sequence and presenting it as a probe for recall to the memory store. During each experiment, one set of vocabulary items was generated and each of the contexts reinstated and presented sequentially in order to generate a retrieved item for each serial position. This process was repeated for each of the 20 sets of context vectors using the same set of vocabulary items and the retrieved items were averaged accordingly. Finally, each of the retrieved items was compared with each of the items in the vocabulary by taking the cosine between the two. The vocabulary item which was most similar to the retrieved item was recalled as the output for that serial position. In order to average the results, a new set of vocabulary of items was generated and the training and recall process repeated for each of the sets of context vectors.

The context distinctiveness was controlled by setting the inter-context spacing between items to three simulated steps. Twenty sets of context vectors were used and the results were averaged for 1000 different sets of vocabulary items.

Results

The serial position curve generated as a result of this simulation is presented below in figure 5.15. The curve exhibits the typical bowed shape with best performance for the items occupying the first and last serial positions. Performance in both of these serial positions is at 99% correct, dropping to 70% in each of the remaining serial positions (except for the third position, where performance drops to 69%).

Figure 5.16 illustrates the distribution of order errors, the transposition gradient, which accompanies the serial position curve of the previous figure. This figure illustrates clearly that each item is recalled maximally in its target serial position. For example, if we consider the items recalled in the third position. The third item is recalled correctly

67% in the third serial position. However, for 15% of trials both the second and fourth items are recalled erroneously in this position. In contrast, the sixth item is recalled 99% of trials in the sixth serial position.

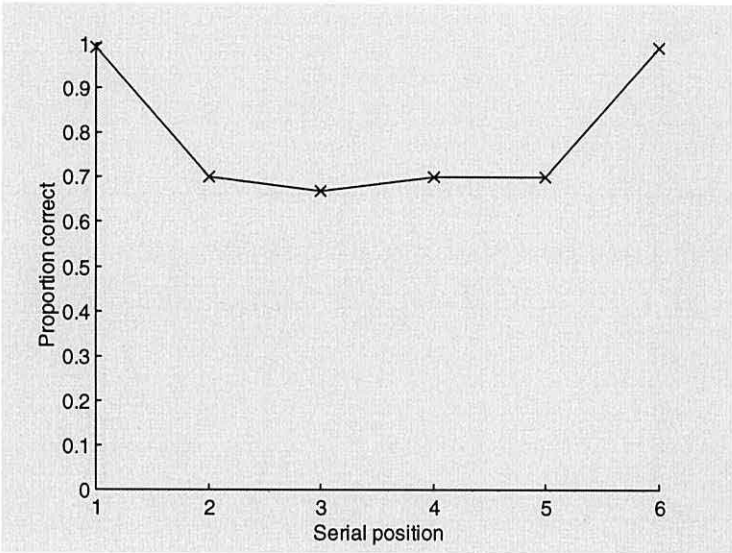

Figure 5.15 *Serial position curve for basic OSCAR network*

Discussion

These results clearly illustrate that even this most minimal implementation of the OSCAR architecture is capable of producing serial ordered behaviour. The serial position curve is bowed with a high degree of first item primacy and last item recency. However, there is no extended primacy effect (i.e. extended over the first two or three items) and the last item recency is greater than was observed empirically (cf. Jahnke, 1963; Baddeley, 1968) for visually presented stimuli.

These effects are attributed to edge effects (cf. Page & Norris, 1995) resulting from confusability between the context vector cues. More precisely, when the first (or last) context vector is presented as a cue for recall, confusion can only occur between contexts in one direction: towards the centre of the sequence. However, when the third or fourth context is presented as the cue for recall, it may be similar with those contexts occupying serial positions in *either* direction towards the ends of the list. In this manner, more item order errors are expected to occur in the central serial

positions than in the end positions. If this is the case, a bowed serial position curve should be the natural consequence.

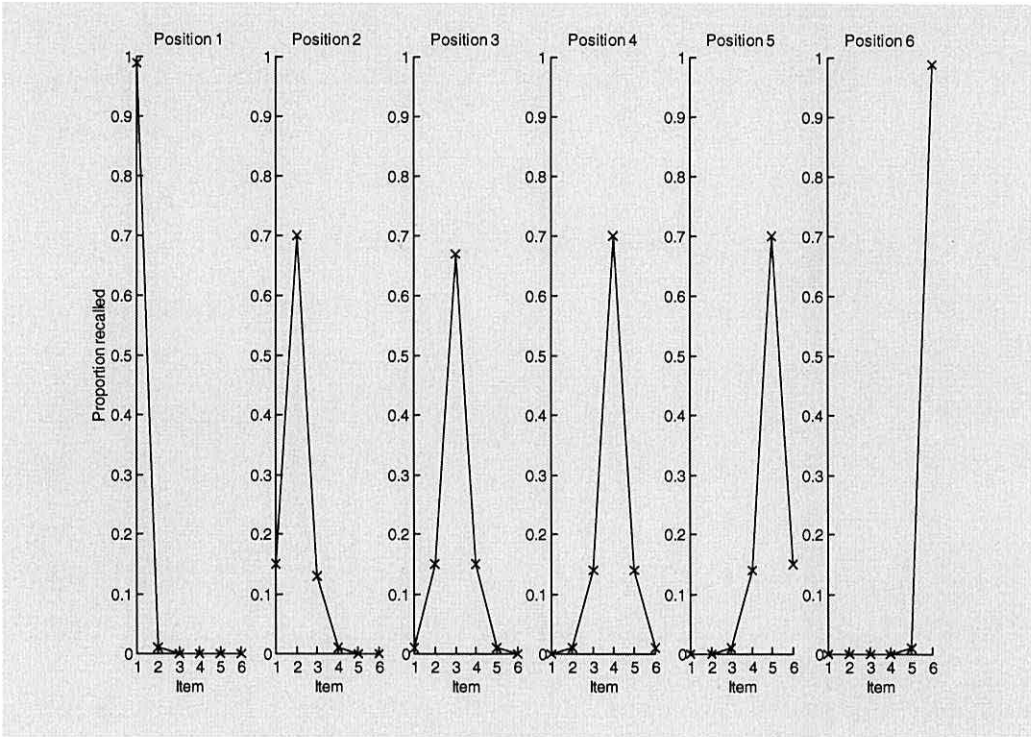

Figure 5.16 *Distance functions for basic OSCAR network*

In fact, this prediction is supported by the distance function (figure 5.16) which illustrates how more order errors occur in the central list positions. This finding is in accordance with a wealth of empirical data (e.g. Estes, 1972; Healy 1974; Henson, Norris, Page & Baddeley, 1996).

However, although it is clear that OSCAR is capable of producing realistic serial order performance, the model does not possess the ability to fit the range of empirical results we require. In order that this may be attempted, a number of free parameters must be added to the network. These are introduced in the following section and explored in detail in the following chapter.

5.7 Additional parameters

Thus far we have introduced the underlying mechanism employed by OSCAR for learning and recall of item-context associations. Although this model performs serial order recall tasks adequately, the serial position curves generated possess symmetric primacy and recency effects (figure 5.15). They lack the extended primacy and small last item recency typical of serial position curves obtained for sequential recall of visually presented stimuli. Therefore, in order to provide a closer match of empirical data using OSCAR, further parameters must be introduced to the system.

In the first instance, two free parameters are introduced: λ , the learning rate and δ the weights decay rate:

$$\hat{\mathbf{M}}_k = \delta \hat{\mathbf{M}}_{k-1} + \lambda \mathbf{i}_k \mathbf{c}_k^T \tag{5.14}$$

The first parameter is the weights decay rate. This introduces recency to the model by degrading the memory trace of earlier list items.

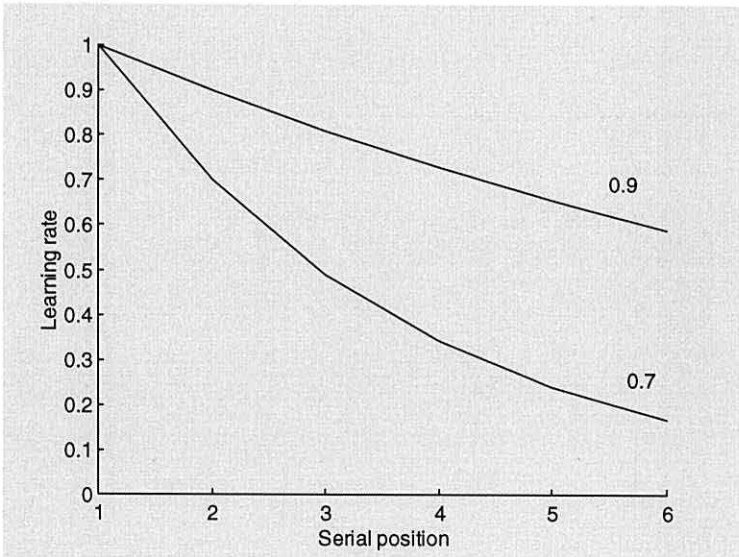

Figure 5.17 *Non-linear learning rate decaying with serial position, i (Lrate=0.9^{i-1} and Lrate=0.7^{i-1})*

The second parameter introduced here is the learning rate. The learning rate effects the strength with which each new item-context pairing is stored in the memory trace. Typically this value is also high, such as 0.9, and constant for each serial position. However, in a number of experiments described here, we use a learning rate term that decreases exponentially with serial position (figure 5.17). This introduces primacy and reflects the intuition that later list items are progressively less surprising or attention-demanding than earlier items.

A number of other parameters are introduced into the system in order to improve performance and are discussed in the next chapter. In order to add output interference, noise is added to the weights matrix during recall. The effect of adding noise to the matrix during the intervals between each item being presented is also considered. The addition of a noise threshold during the retrieval stage supplements the order errors that occur naturally in the model with item and omission errors. Furthermore, a number of inhibitory processes are investigated.

5.8 Summary

In this section, an oscillator-based associate recall model of serial ordered learning and recall, OSCAR, has been introduced and its basic architecture outlined. The notion of a reinstatable temporal signal with a non-repetition property and a specific similarity function has been introduced and a method for generating a suitable signal described. By associating novel distributed vectors of features with successive context vectors, it is possible to produce serial ordered learning and recall. This is reflected in the bowing of the serial position curve and the natural distribution of order errors about target serial positions. However, without additional parameters, the model is unable to fit sufficient of the empirical data we wish to address. In the next chapter, we examine how the introduction of these parameters influences the behaviour of the model.

CHAPTER 6

Investigating OSCAR parameter space

6.1 Introduction

At the end of the previous chapter, it was shown that the simplest implementation of the OSCAR architecture was capable of producing a bowed serial position curve and a natural distribution of order errors. We also explained the need for two important free parameters: the constant weights decay rate, δ, and the variable learning rate, λ, that may decrease exponentially with serial position. In the following chapter, a summary of the effects of varying OSCAR's free parameters, either independently or in combination, is presented. The purpose of this is to gain a computational understanding of OSCAR's behaviour. The psychological data is addressed in chapter 7.

6.2 Simulation 11: *Reducing the distinctiveness of the context vectors*

Introduction

It was suggested in section 5.5.2 that the ability to discriminate between learned items might depend upon the distinctiveness of the context vectors. To review, we predicted that the use of "less distinct" contexts would result in more item errors as the recall cues would be easily confused while the use of "highly distinct" contexts would result in a high degree of performance. The distinctiveness can be controlled by manipulating the degree to which the oscillators evolve during the simulated time interval between each being selected. For example, if the oscillators are allowed to evolve for a long period then the contexts will be "highly distinct". However, if the contexts are selected rapidly and the oscillators prevented from evolving far during each inter-context interval, then the resulting set of contexts will be "less distinct". In

practice, the distinctiveness is controlled by selecting context vectors from a sequence and controlling the gap between each context, the *jump*.

Method

The method employed in the following simulation is the same as that outlined for simulation 10. However, during the first part of the simulation, the distinctiveness between the contexts was varied by selecting contexts separated by distances of either two, four or six steps. Performance for each condition is examined in terms of the serial position curve. In the second half of the simulation, the distinctiveness was manipulated by increasing the spacing between context vectors from one step through to ten. For each case, the mean proportion of items recalled correctly in their correct serial position was recorded.

In both simulations, twenty sets of context vectors were used and the results were calculated using the average of 500 different sets of vocabulary items. No inhibitory processes, learning rate or decay rate parameters were introduced during this simulation and as such the results reflect the performance of the basic OSCAR architecture.

Results

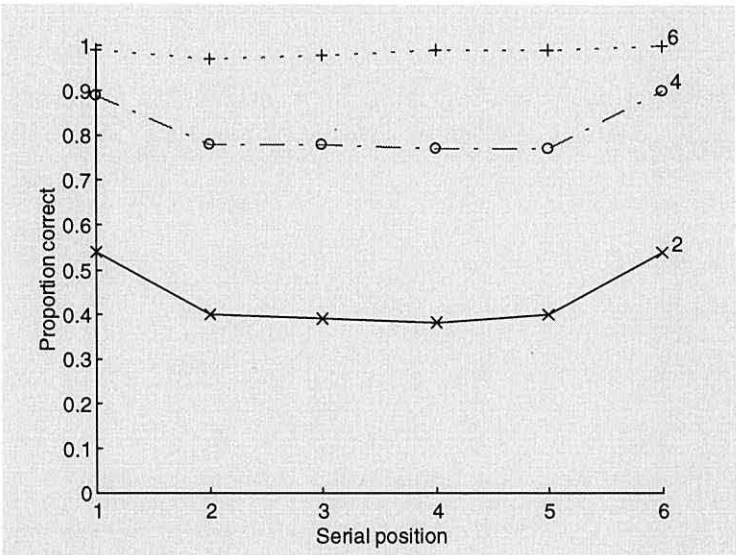

Figure 6.1 *Serial position curves for six item lists as the inter-context spacing is increased from two to six steps*

Figure 6.1 illustrates the results of reducing the distinctiveness of each set of context vectors by selecting closer spaced sets of context vectors. Performance when the spacing is large (six steps) is nearly 100% for each of the six serial positions. However, as the contexts become "less distinct" and the spacing is reduced (to only two steps) so performance drops to approximately 45% for the first and last serial positions and approximately 40% for the central list positions.

Figure 6.2 illustrates the mean proportion of items recalled correctly for six item lists as a function of increasing distinctiveness, presented as increasing inter-context spacing. Mean recall improves from approximately 25% when the inter-context spacing is unity, to approximately 83% when the inter-context spacing is four steps, to a ceiling value of approximately 100% for inter-context spacing of seven steps or more.

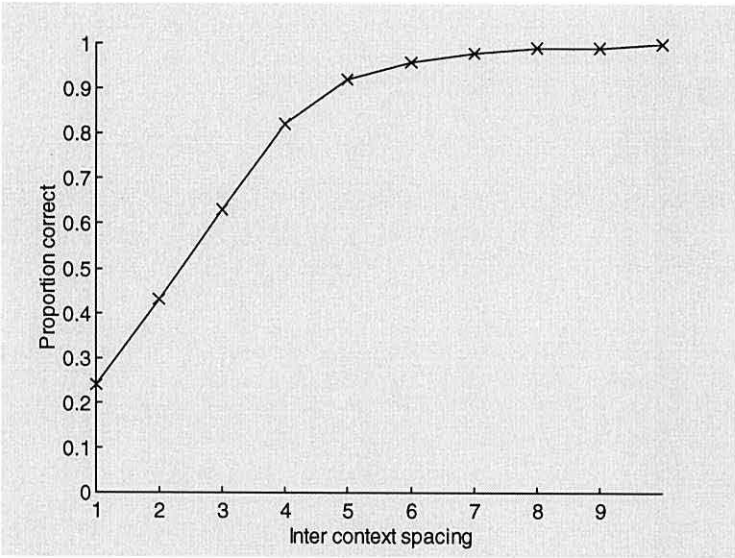

Figure 6.2 *Mean proportion correct recalls for increasing inter-context spacing*

Discussion

The results of this simulation are clearly illustrated in figures 6.1 and 6.2. As predicted, when the inter-context spacing and distinctiveness between the contexts is increased, so recall performance improves. Figure 6.2 illustrates that recall performance increases towards the 100% ceiling level as the distinctiveness increases. It is clear from these results that an inter-context spacing of four or five simulated

time steps is sufficient, as at this level the serial position curve is adequately bowed and performance is sufficiently high to produce realistic levels of recall.

In fact, it is important to consider not only the effect of inter-context spacing on the models ceiling performance, but also the effect that the capacity of a Hebbian network for accurate storage and retrieval of non-orthogonal data may have on the models behaviour. A Hebbian network can only store and recall the associations of n n-dimensional vectors and orthogonal contexts with absolute accuracy[11]. In the case of OSCAR, where the item and context vectors are non-orthogonal, performance is much reduced and hence large dimensionality vectors have to be used.

In the light of the simulation described here, in subsequent simulations a more distinct set of contexts will be employed in order to improve performance. In simulation 10, an inter-context spacing of three steps was used, however for all future simulations, the comparison between the refined model and the basic architecture will use an inter-context spacing of four unless stated otherwise.

However, although OSCAR can produce serial position curves without the use of any free parameters, it is only by introducing additional parameters to the model that its performance may be fitted to empirical data.

6.3 Effect of decay rate and learning rate

There is much evidence to suggest that memory traces decay over time (e.g. Conrad, 1960) and the majority of models of short-term memory implement the decay of memory contents in some manner by scaling the current contents of memory prior to each new association being combined with the current contents of memory. For example, the forgetting parameter, α, of TODAM (Lewandowsky & Murdock, 1989) which implements forgetting through decays by premultiplying the contents of the memory vector before the next item and association are added to the composite memory trace. Without weights decay, each weights matrix element could grow

[11] Imagine each context is a binary vector with zeros occupying n-1 elements and a one in the element that corresponds to the list position of the stimuli item.

towards infinity. Furthermore, it is also commonplace to weight the strength with which each new item is stored in the memory trace by the use of a learning rate parameter (cf. the item and order weighting parameters, γ and ω, of TODAM (Lewandowsky & Murdock, 1989)). The effect of manipulating OSCAR's learning rate and weights decay parameters both separately and in combination is reported in this section.

6.3.1 Simulation 12: *Introducing a decay rate parameter*

Introduction

In the following simulation, a "decay rate" parameter which varies over the range 0 to 1, δ, is introduced in order to scale the composite memory trace before each new association is added to the store (see equation 5.14). It is anticipated that this parameter will introduce an extended recency effect to the serial position curve as it will degrade the earliest associations. Thus, earlier items will be harder to restore in comparison with those occupying the last serial positions.

Method

The procedure employed during the following simulation was identical to that described for simulation 10. However, in contrast, the inter-context spacing between items was equal to four. Furthermore, a "decay rate" parameter was introduced during the learning stage. This scales the contents of the Hebbian memory matrix prior to adding the next association to the memory. This was assigned a value of 0.9 which corresponds to a decay of memory of 10% before every association.

Results

Figure 6.3 illustrates the effect of introducing a decay rate parameter of 0.9 to the OSCAR architecture and contrasts it with the performance of the most basic OSCAR architecture (with inter-context spacing of four).

Performance for the first item is almost exactly the same as for the parameter free model at around 90% correct. However, for each of the subsequent items, performance improves until recall is almost 100% for the sixth item.

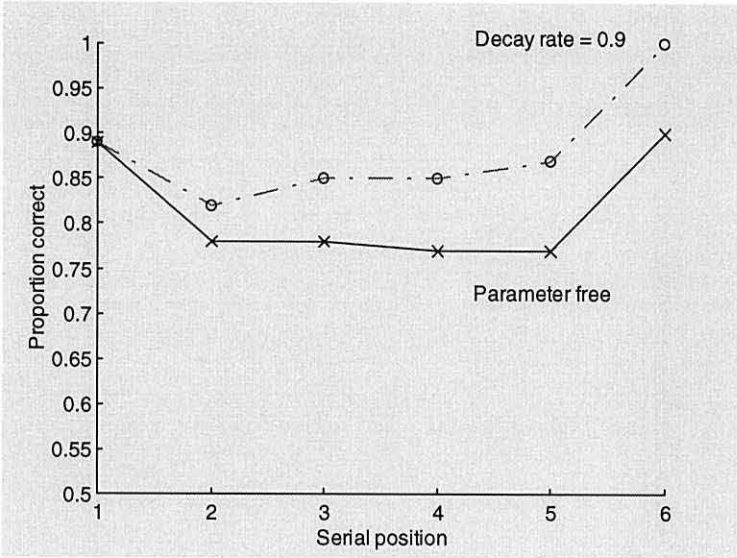

Figure 6.3 *Serial position curve with decay rate parameter*

Discussion

What is immediately evident from figure 6.3 is that performance for the decay condition is better overall than for the parameter free condition. The first item has not suffered even though the association between it and the first context has been scaled repeatedly. However, the relative first item primacy has been reduced due to the increase in performance for the second, and all the subsequent, items. The fact that performance is as good for the first item with inclusion of the decay rate parameter as it is without is attributed to the fact that the associative and recall mechanisms are robust enough to be able to recognise that the first item is associated with the first context. It would be reasonable to predict that if the contexts were less distinct (i.e. the inter-context spacing was reduced) or the decay rate parameter reduced, then the performance for the first item would be reduced further. The second point that should be observed is that the recency component of the last item over the second to last item remains approximately constant for both conditions.

However, the question remains as to why performance for the decay condition, although as predicted (i.e. better for later than earlier items), is better overall than for the parameter free condition. We would suggest that as the learning and recall process is robust, once the first item has been recalled, the likelihood of recalling the next item increases as it will be less decayed than the previous item. Increasing the magnitude of the decay would once again decrease performance overall.

Therefore, although the serial position curve contains a primacy and recency component, the primacy component is not similar to that which we would anticipate for a immediate recall task for visually presented stimuli. In order to address this, the inclusion of a learning rate parameter is considered in the following simulation.

6.3.2 Simulation 13: *Introducing a learning rate parameter*

Introduction

In the following simulation, a non-linear learning rate parameter, λ, which varies over the range 0 to 1, is introduced in order to scale the strength with which each new association is added to the composite memory store (see equation 5.14). As the learning rate will be greatest for the first association, decreasing non-linearly for each of the subsequent associations, it is anticipated that it will introduce primacy to the serial position curve by reducing the strength with which the later items are combined to the memory trace. However, this may be at the expense of overall performance.

Method

The procedure employed during the following simulation was identical to that described in experiment 10 but with the following differences. The context vector distinctiveness was controlled by setting the inter-context spacing to four. The decay rate parameter described previously was excluded from the current simulation. However, a non-linear learning rate parameter was introduced and took a value of unity for the first association and then 90% of the previous value for each of the subsequent associations (cf. figure 5.17). Performance is compared with a parameter free version of OSCAR with identical context distinctiveness.

Results

The effects of introducing a non-linear learning rate parameter to the OSCAR architecture are illustrated in the serial position curve of figure 6.4. As before, performance is unchanged for the first item but decreases to a low of approximately 65% for the fifth item. Performance improves slightly for the last item with a four percent last item recency effect, but, overall, falls below that of the parameter free version.

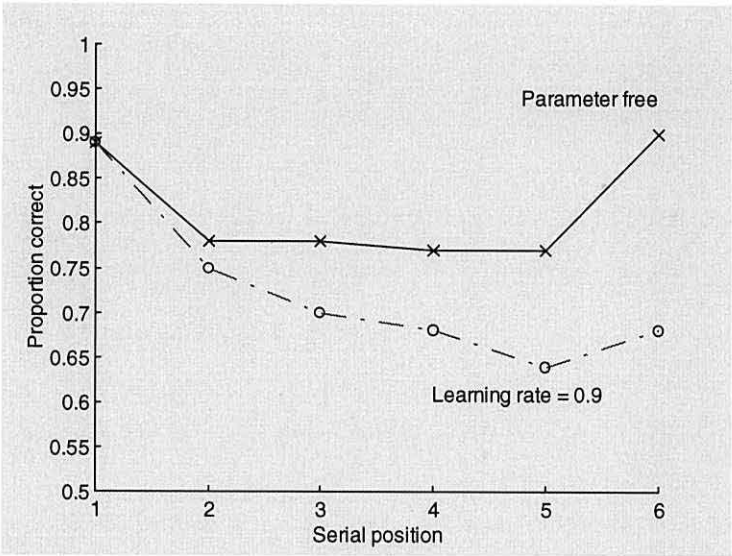

Figure 6.4 *Serial position curve with learning rate parameter*

Discussion

It is evident from figure 6.4 that the reverse effect to that demonstrated by the decay rate parameter (figure 6.3) occurs. In the present case there is a primacy effect extending over the first four items and a reduction in last item recency to four percent.

As in the previous simulation, performance for the first item is unaltered by the inclusion of this parameter. In this case, however, this is exactly as would be expected as the learning rate for the first item is identical to that for the parameter free condition. As the first association is in no way degraded, we would expect performance to reflect the maximum possible for a network with normalised distributed stimuli and contexts. It should also be observed that unlike the previous simulation, the mean performance is worse than the parameter free condition.

Having outlined the effects of adding decay and varying the learning rate across serial position, the next step is to combine these two parameters and vary the degree to which each controls the performance of the serial order recall task. Therefore, the following simulation examines the effects of employing a combination of both the learning rate and weights decay parameters during performance.

6.3.3 Simulation 14: *Combining decay and learning rate*

Introduction

In the following simulation, the non-linear learning rate, λ, and the weights decay parameter, δ, are introduced in different combinations to the model. The simulation consists of three different conditions: in the first the learning rate will dominate; in the second, the weights decay parameter will have the greatest influence; in the third condition, both parameters will have similar influence on performance.

It is anticipated that the subtle combination of the two free parameters should be sufficient to introduce the extended primacy and last item recency desired of the model without degrading the overall mean performance.

Method

The procedure employed for the following simulation is identical to that described in experiment 10. However, for each of the three conditions the learning rate and weights decay parameters are varied. In the first condition, the learning rate parameter is assigned a value of 0.7 and the weights decay a value of 0.9. In the second condition, the learning rate parameter is assigned a value of 0.9 and the weights decay a value of 0.7. In the third condition, both the learning rate parameter and the weights decay are assigned values of 0.9. In each condition, the context distinctiveness is controlled by setting the inter-context spacing to four. Performance in each condition is compared with the basic OSCAR network.

Results

The results for the three conditions are illustrated in figure 6.5. In the first condition, the learning rate influences the serial recall performance. There is extensive first item primacy (30%), however performance decreases for each subsequent item and all recency is eliminated. In the second condition, the opposite is found as all primacy is eliminated while performance increases to almost 100% in the last serial position with a 10% last item recency effect. In the third condition, where the learning rate and weights decay parameters are assigned identical values, the subtle influence of each parameter upon the serial position curve is evident.

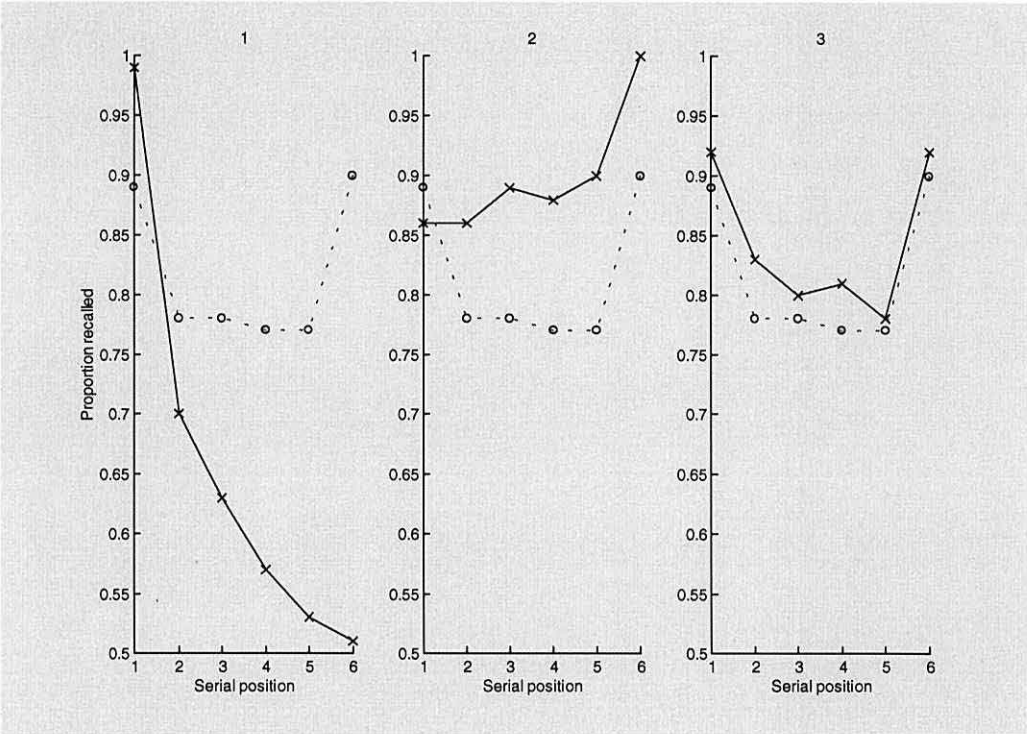

Figure 6.5 *Serial position curves illustrating the combined effects of the non-linear learning rate and weights decay rate (Condition 1: λ=0.7, δ=0.9. Condition 2: λ=0.9, δ=0.7. Condition 3: λ=0.9, δ=0.9. Dotted line: λ=1, δ=1.)*

Performance in this condition is better than for the parameter free condition in each serial position. There is an extended primacy effect with approximately 9% first item primacy. The last item recency effect is considerable (approximately 14%) while the curve replicates the asymmetry typical of the empirical data for immediate serial order recall of visually presented stimuli (e.g. Jahnke, 1963; Baddeley, 1968).

Discussion

Figure 6.5 (condition 1) illustrates that with such a steep learning rate, it is the learning rate that proves responsible for the general shape of the curve. There is an excellent first item primacy effect. The learning rate benefits the item in the first serial position as it is stored in the memory trace with a weighting of one, whereas the next item weighted by a factor of 0.7 before storing, then by 0.49 for the next item and so forth for each of the remaining serial positions. The net result is that the first association is by far the strongest presented to the memory matrix and hence the first item is recalled 99% of the time. However, this is at the expense of any of the recency provided by the decay rate term.

The overall shape of the curve illustrated in figure 6.5 (condition 2) is provided by the decay rate. Comparing this figure with that of figure 6.3, it appears that performance is almost identical to that reported by the use of a decay rate term alone (Simulation 12). However, in the present simulation the small degree of primacy apparent in figure 6.4 is eliminated due to the reduction in the decay rate parameter. Clearly the decay rate parameter has a very large influence on the model and should therefore be assigned large values (e.g. δ=0.9) in order to ensure that the primacy effect is not eliminated at the expense of improved performance of the latter half of the list.

Finally, the results of the third condition illustrate that when the learning rate and decay rate parameters are combined and used more subtly, the effect of each can be controlled to produce a serial position curve with both an extended primacy effect, last item recency and the asymmetry typical of the empirical data (e.g. Jahnke, 1963; Baddeley, 1968) for immediate recall of visually presented stimuli. However, here we do no attempt to reproduce the empirical data exactly as it is unclear how these parameters will interact with others, such as inhibition.

6.4 Effect of output inhibition

6.4.1 Introduction

The data presented so far has all been generated in the same manner: an output vector, generated by probing the memory matrix with one of the context vectors presented during learning, is compared with a vocabulary of items, some, but not necessarily all of which, were presented during the learning stage. The model calculates the cosine between the retrieved vector and each of these vocabulary items. The vocabulary item that results in the greatest cosine (i.e. closest to unity, where unity would signify that the item was identical to the target item) is recalled. However, given such a simple model, there is a tendency for many items to be recalled in more than one serial position. This results in more errors than would be expected from the model and an unnatural error distribution.

One solution to this problem is the introduction of a number of simple inhibitory processes during recall (Burgess & Hitch, 1992, 1996; Henson, 1996; Houghton, 1990, 1994a; Page & Norris, 1995).

Houghton (1990; 1994a) describes an opponent processing system in the "competitive filter" of the competitive queuing model. A layer of opponent nodes, driven by corresponding item nodes, first select the most active item node, then inhibit laterally in order to suppress the other outputs. Finally, they inhibit the most active item node, permitting competition for the next output. Such a process necessitates the need for a "doubling" mechanism that allows sequences containing repeated items to be recalled (Houghton, Glasspool & Shallice, 1994). Henson, Norris, Page and Baddeley (1996) observe that where repeat errors occur, they are usually consist of early items being repeated at the end of the list. Attempting to account for this effect with the primacy model, Page and Norris (1995) suggest a decaying inhibitory mechanism such that the likelihood of a repeat error increases with serial position. A similar process is outlined by Burgess and Hitch (1996) in order to produce a more localised distribution of order errors.

In the following section, the effects of extending the OSCAR retrieval stage with one of two simple inhibitory processes is considered. The first is a simple process which prevents the same item being recalled in adjacent serial positions. This may be thought of as a simplistic representation of the decaying inhibitory mechanism outlined by Page and Norris (1995). The second process is a more powerful mechanism which prevents any item being recalled twice in any one sequence. Nairne and Neath (1995) observe that the sampling without replacement of TODAM (Lewandowsky & Murdock, 1989) is effectively a no-repeated-items inhibitory mechanism.

6.4.2 Simulation 15: *Simple inhibitory process during recall*

Introduction

The first inhibitory process investigated here is a rule which prevents the successive recall of any one item in neighbouring positions, such as with the incorrect recall of

the list *ABCDEF* as *ABCDDF*. In this case, the second *D* would be prevented from being recalled, and the next most similar item retrieved. This rule is particularly important because the majority of the order errors produced by the model in its most basic form are due to the similarity of the cue context with those contexts in the immediately adjacent serial positions. By inhibiting the recall of any one item in successive serial positions, recall should be dramatically improved but should still allow repeat errors to occur later in the list.

For example, if the first item is recalled accurately the majority of times, then the second item is more likely to be recalled correctly because the recall of the first item in the second serial position will be prevented. This improvement may be propagated throughout each serial position. However, the converse is also true, in that if the first item is mistakenly recalled in the first serial position as the second item, then in the next serial position it will be impossible to retrieve the correct item even if it is the most similar when compared with the recalled item. This will prevent the second item being retrieved correctly. By the third serial position, the model will be able to recover, unless the item retrieved in the previous position was in fact the third item.

When an item is erroneously recalled in a serial position immediately before its target position (e.g. item *p* recalled in position *p-1*), most probably the item that will be recalled next will be the one that should have been recalled correctly in the current serial position (e.g. item *p-1* recalled in position *p*). Page and Norris (1995) refer to this property as *fill-in* and is a feature of the primacy model not shared by TODAM (Lewandowsky & Murdock, 1989) or the Burgess & Hitch (1992; 1996) model. Fill-in occurs in the Page and Norris (1995) primacy model as a result of the bias towards earlier list items built into the primacy gradient.

In the following simulation we examine the effect of the addition of a simple inhibitory mechanism in order to prevent the same item being recalled in successive serial positions.

Method

The following experiment implements the same procedure as was described in experiment 10 with the following additions. For both the experimental condition and the control (parameter free) condition, a more distinct set of context vectors is employed. This is implemented by increasing the inter-context spacing to four. Furthermore, a simple inhibitory process was added to the retrieval stage. Simply, when the most similar item was retrieved from the lexicon of recallable items, if it was identical to that recalled in the previous serial position, it was inhibited and the next most similar item selected instead.

Results

Figure 6.6 illustrates the effect that combining this rule with the basic OSCAR architecture has on the serial position curve. It is clear that overall, performance has decreased, although the first and second items are almost unaffected by the inhibitory mechanism (this is not surprising as the rule can only apply to the second and all consecutive items). Performance decreases for subsequent items and reaches a trough of 69% in the fifth serial position. There is a last item recency effect of 11%.

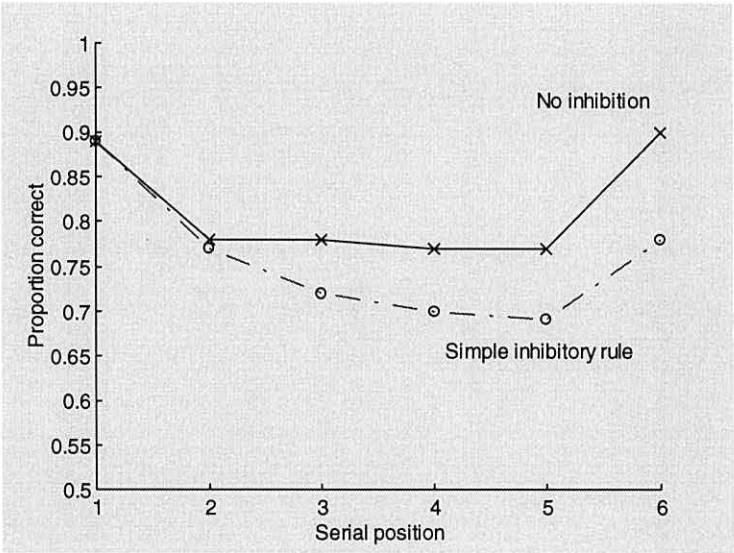

Figure 6.6 *The effect of a simple inhibitory process during recall*

Figure 6.7 illustrates the distance function that accompanies the serial position curve of figure 6.6. It is evident that the first item was recalled in the first, and correct,

serial position approximately 89% of the time. The second item was recalled 10% of trials in this position, while the sixth item was not recalled at all in this position.

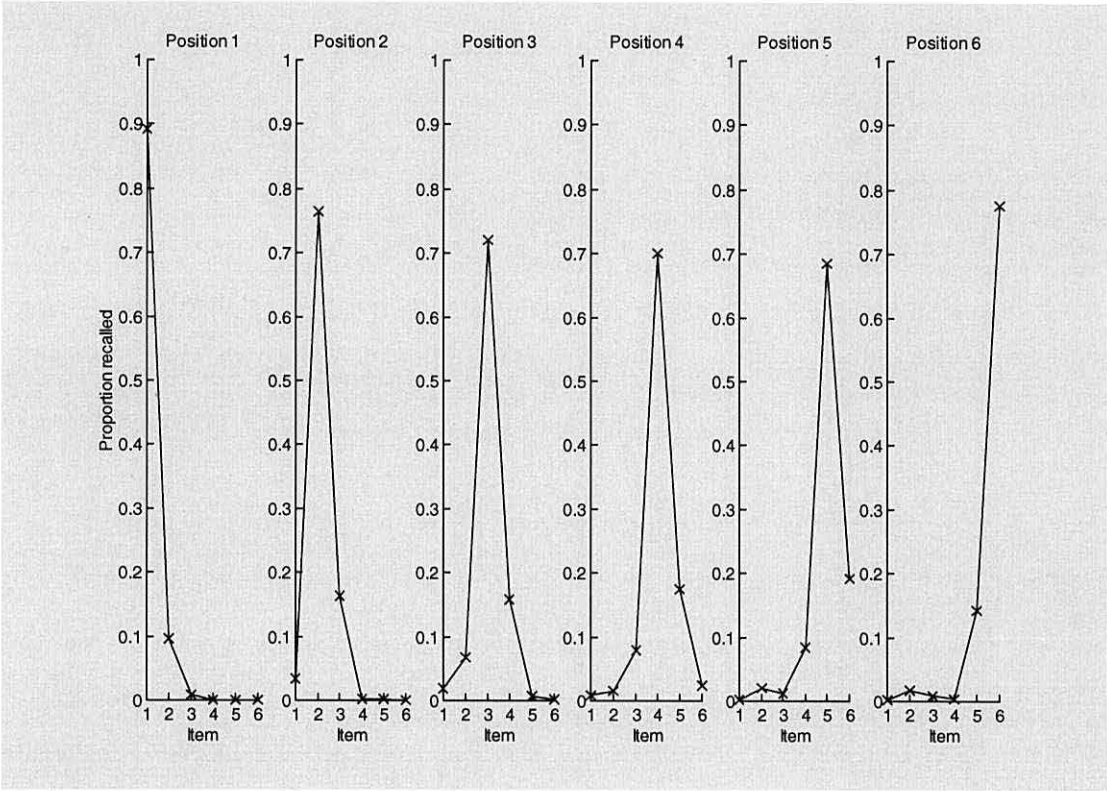

Figure 6.7 *Distance functions illustrating the effect of a simple inhibitory process during recall*

The distance functions also illustrate that each item is the most active in its own, target, serial position. For example, consider the fourth serial position in figure 6.7. The first and second items are only recalled in 1% or 2% of trials. However, the third item is recalled in 8% of trials, the fourth item maximally in 70%, the fifth in 18% and the sixth in only 2%. Note that plotting the proportion of correct recalls in each target serial position reproduces the serial position curve of figure 6.6.

Discussion

These results would appear to confirm the suspicion that performance on later items suffers by the inclusion of a simple inhibitory rule. The serial position curve bears a striking similarity to the curve generated with the inclusion of a learning rate parameter alone (cf. figure 6.4). However, analysis of the distance function provides a more thorough account of why the curve is generated in this manner.

The first item is recalled correctly 89% of the time in the first serial position and the simple inhibitory mechanism prevents it being recalled in the second position. However, the second item benefits little from this, and is recalled slightly less than during recall without the rule. It is by examining the third and fourth serial positions, however, that the reason for the gradual reduction in performance becomes apparent.

Item three is not recalled in the first serial position (due to the lack of similarity between the third and first contexts), however it is recalled almost 17% of trials in the second serial position (i.e. before the target, third, serial position). This is significant as the third item is recalled a smaller proportion of times in the fourth serial position, approximately 8% of the time. A similar effect is found for the fourth item. This would suggest that when errors are occurring earlier in the list where OSCAR has failed to select the appropriate item for recall, it chooses to recall the next item as recall of the previous item is prevented. This is reflected in the asymmetry of the order errors about the target serial position (cf. figure 5.16). Clearly, as items are being recalled erroneously before their target serial position, so performance in the target positions is reduced. For this reason, OSCAR produces a poorer serial position curve than that produced without the inclusion of the inhibitory rule and although fill-in (Page & Norris, 1995) occurs, it is only in a small proportion of trials.

However, accepting that performance overall is reduced, the serial position curve retains the desired extended primacy and last item recency effects of the empirical data (e.g. Jahnke, 1963; Baddeley, 1968), and now possesses a more natural error distribution with fewer duplicate item errors occurring in neighbouring serial positions.

6.4.3 Simulation 16: *Preventing item replication during recall*

Introduction

A second and more powerful inhibitory rule is one which prevents repeated items occurring *anywhere* in the serial position curve. For example. during recall of the list *ABCDEF*, errors such as *ABCD<u>D</u>F* and also *ADC<u>D</u>EF* would be prevented (where the underlined item would be prevented).

The fact that a list must not contain a repeated item is often stated explicitly during empirical procedures (e.g. Baddeley, 1968, experiment 5). Nairne and Neath (1994) observe that the use of the "sampling without replacement" mechanism in TODAM (Lewandowsky & Murdock, 1989) can be likened to the use of a rule preventing repeated items occurring in a recall sequence. They observe that such a process eliminates the recency portion of the serial position curve (Nairne & Neath, 1994, figure 1). We would anticipate that a similar effect would be demonstrated here using OSCAR.

Method

The following experiment executes the same procedure as was described in experiment 10 with the following additions. For both the experimental condition and the control (parameter free) condition, a more distinct set of context vectors is employed. This was implemented by increasing the inter-context spacing to four.

Furthermore, if the item selected from the lexicon as providing the closest match to the retrieved item was identical to any of the items recalled in *any* previous serial position, then recall of that item was prevented and the lexicon item with the next greatest cosine (that had not been recalled previously) was recalled instead.

Results

The results of this simulation are illustrated in the serial position curve of figure 6.8 and the corresponding distance function of figure 6.9.

As before, recall of the first two items appears to be unaffected by this inhibitory rule (again, this is as would be expected given that the rule does not affect the first item and, given the accuracy of recall of the first item, influences the second item very little also). However, recall performance reduces for each of the subsequent serial positions, dropping to a low of approximately 63% in the fifth serial position. Most noticeably there is only a 1% last item recency effect.

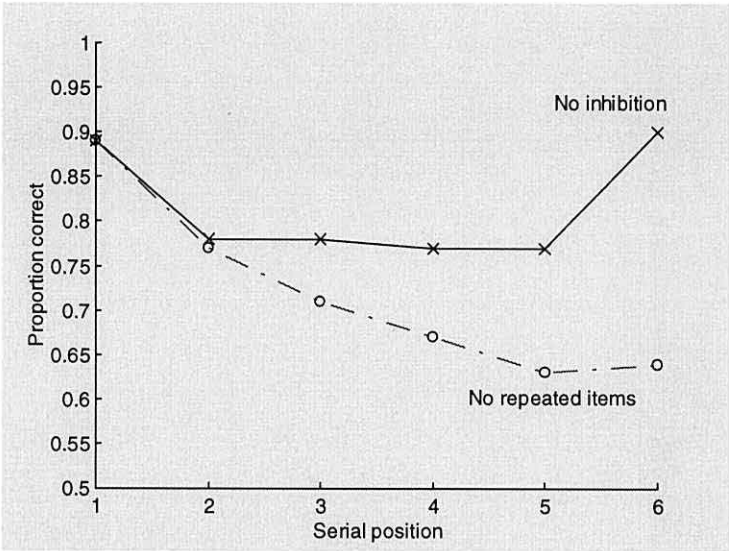

Figure 6.8 *Introducing an inhibitory process preventing repeated item errors*

The distance function of figure 6.9 illustrates the effect of the inhibitory process. Estes' (1972) observation that distance functions appear symmetric about the central list position clearly does not apply in this case. Performance in the first serial position is similar to the parameter and inhibition free condition. The first item is recalled the most and any order errors occur with items immediately adjacent to the target item. However, this is not the case in the sixth serial position, where order errors occur evenly with each of the five previous items occurring in equal proportions in the sixth serial position. Furthermore, the inhibition is reflected in the asymmetry of order errors in the central list positions. Previously, order errors occurred in approximately equal proportions about the target item.

However, in the current condition, clearly order errors occur in greater proportions with items that are yet to be recalled in their target serial positions. This is illustrated most clearly by the errors that occur in the fourth and fifth serial positions where the early items are recalled in less than 1% of trials. However, even in the list positions immediately prior to the target list position, the errors only occur in small proportions (e.g. the third item is recalled only 4% of trials in the fourth serial position, while the fifth item is recalled 22% of trials).

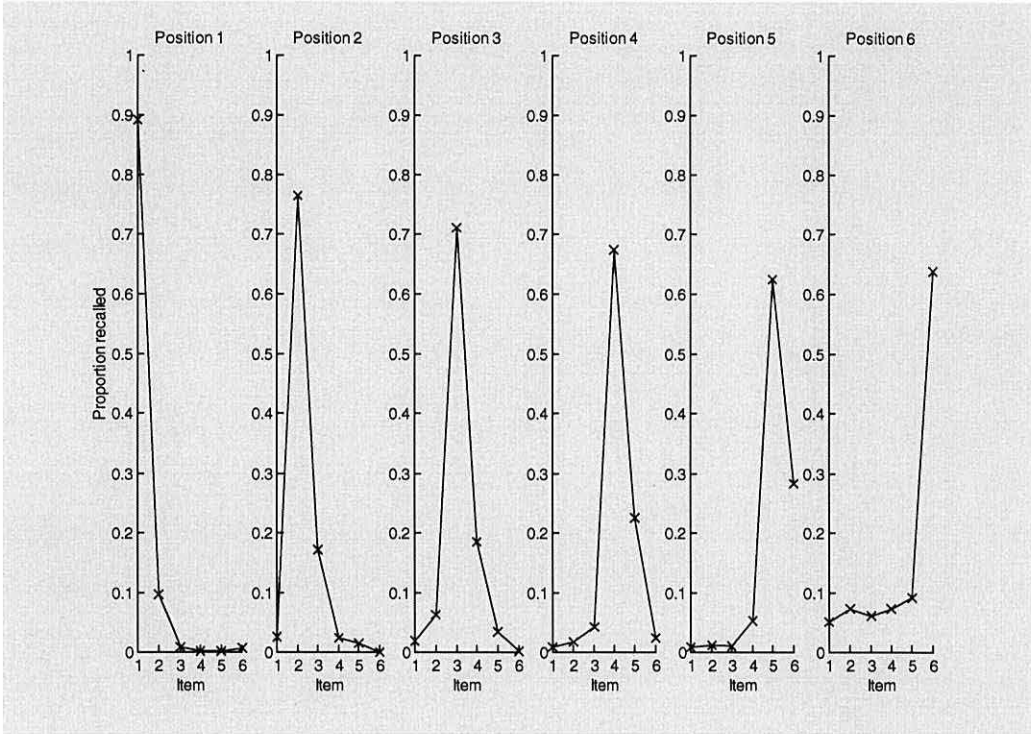

Figure 6.9 *Distance functions illustrating the effect of an inhibitory process preventing repeated item errors during recall*

Discussion

It is clear from the results that, in accordance with Nairne and Neath (1994), the inclusion of the rule preventing repeated item errors during recall eliminates the recency portion of the serial position curve (figure 6.8). This effect may be explained by close examination of the accompanying distance function of figure 6.9.

The inhibitory process preventing items being recalled more than once in any one trial forces errors to occur in a number of different ways. It is immediately evident that there is now a more pronounced asymmetry within the distance function: items are more likely to be recalled erroneously before their target serial position than after. It is also evident that for each item, once its target serial position has been reached, and the peak recall returned at this point, recall drops to (near) zero for each of the remaining serial positions with the exception of the sixth, and last, serial position. In this case, it is clear that each of the items is recalled approximately 7% or 8% of trials as an *intra-list* intrusion in the sixth serial position.

Given that there is no vocabulary of distractor items (i.e. unlearned items) to choose a response from, and no opportunity to omit to recall an item for any serial position (i.e. all errors are *order* errors), the model is forced to recall whichever item is yet unrecalled by this point. So, if, for example, the model recalled item *B* in the first serial position (in favour of item *A*), there is a good chance that when probed with the context for the second serial position that item *C* may be retrieved because recalling *B* is prevented. From this serial position onwards, the chances of recalling *A* decrease as the contexts corresponding to the latest serial position and the first serial position become less similar. But as more items (correct or otherwise) are retrieved from the vocabulary, so the chances of *A* being recalled in error will increase until, by the final serial position, it has to be recalled as there are no other suitable competitors for that serial position. This accounts for the low performance and lack of recency in this last serial position. Nairne and Neath (1994) observe a similar effect with TODAM (Lewandowsky & Murdock, 1989) and, as Lewandowsky and Murdock use the property to control recency, regard it as unfavourable. Here, we observe that the effect exists and that it can be explained fully by examination of the distance functions that accompany the serial position curve.

Given that the elimination of the recency component of the serial position curve may be accounted for by the lack of alternate lexicon items available for selection in the serial positions where the target item has already been erroneously recalled in a prior serial position, it would be prudent to examine the effect of offering a vocabulary of distractor items to the network during recall.

6.4.4 Simulation 17: *Inhibition and vocabulary of distractor items*

Introduction

In the previous data, it was demonstrated that if there was no lexicon of distractor items available at recall, then, in the presence of an inhibitory process preventing the recall of any item twice, the serial position curve would lose the recency component. In the following simulation where inhibition is once again present, the effect of expanding the lexicon of recallable items to include a number of previously unseen, and unlearned, distractor items, is considered. Given that the lack of recency in the

previous simulation is attributed to the lack of recallable items for the last serial position, we predict that the addition of a lexicon of unlearned distractor items should reintroduce some degree of recency into the serial position curve.

Method

The following experiment uses the same procedure as was described in experiment 10 with the following additions. For both the experimental condition and the control (parameter free) condition, a more distinct set of context vectors is employed. This was implemented by increasing the inter-context spacing to four.

Furthermore, for each set of simulations, when a new stimulus vocabulary was created, so too was a similar set of vocabulary items. This latter set of items, although not presented to the network during learning, was added with those stimuli that were learned and made available as the lexicon of recallable items during the retrieval and deblurring stage.

As with the previous simulation, if the item selected from the lexicon as providing the closest match to the retrieved item was identical to any of the items recalled in *any* previous serial position, then recall of that item was prevented and the lexicon item with the next greatest cosine, that had not been recalled previously, was recalled instead.

Results

The serial position curve for this simulation is presented in figure 6.10 and the accompanying distance function in figure 6.11.

Once again, performance over the first two serial positions is similar to that of the basic, inhibition free, OSCAR model (at 89% and 77% respectively). Performance decreases steadily to a trough of approximately 64% in the fifth serial position. There is a sizeable last item recency effect of 12%.

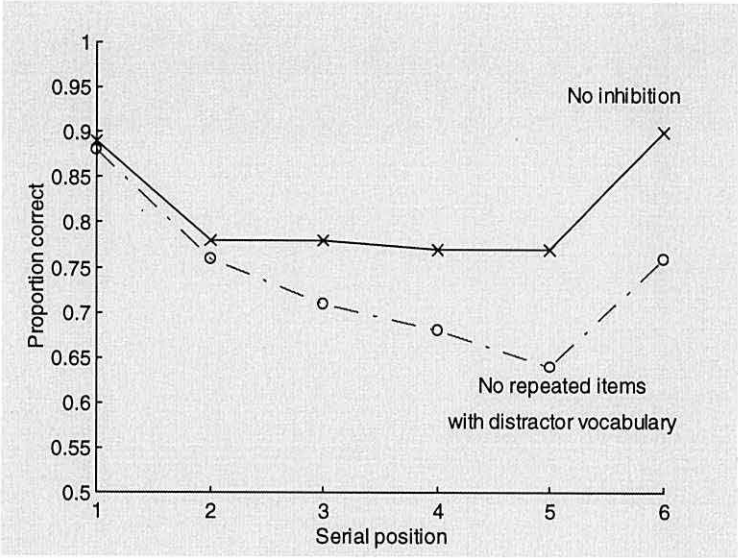

Figure 6.10 *Introducing an inhibitory process prevented repeated item errors with the addition of a vocabulary of distractor items during recall*

The distance function illustrated in figure 6.11 confirms that the asymmetry of the inhibited OSCAR model remains although there are noticeably fewer intra-list intrusion errors occurring in the sixth serial position (cf. figure 6.9). The asymmetry within the distance function is clear. For example, in the fourth serial position we would expect the third and fifth items to be recalled in approximately the same proportions. However, this is clearly not the case as the third item is recalled in only 3% of trials whilst the fifth item is recalled in over 16% of trials. However, it is clear that in contrast to the previous simulation, very few intra-list errors occur in the sixth serial position and this is reflected in an increase in last item recency (cf. figure 6.10).

Discussion

The most striking feature of figure 6.10 when compared with figure 6.8 is that the recency component of the serial position curve has returned with the inclusion of a vocabulary of distractor items at recall.

This behaviour may be explained by once again examining the distance function (figure 6.11). It is evident that items are recalled erroneously more often before, than after, their target list position. However, in order to account for the reinstatement of the recency component and the absence of the cluster of intra-list intrusion errors in

the sixth serial position, we need to consider the distribution of the errors in earlier serial positions.

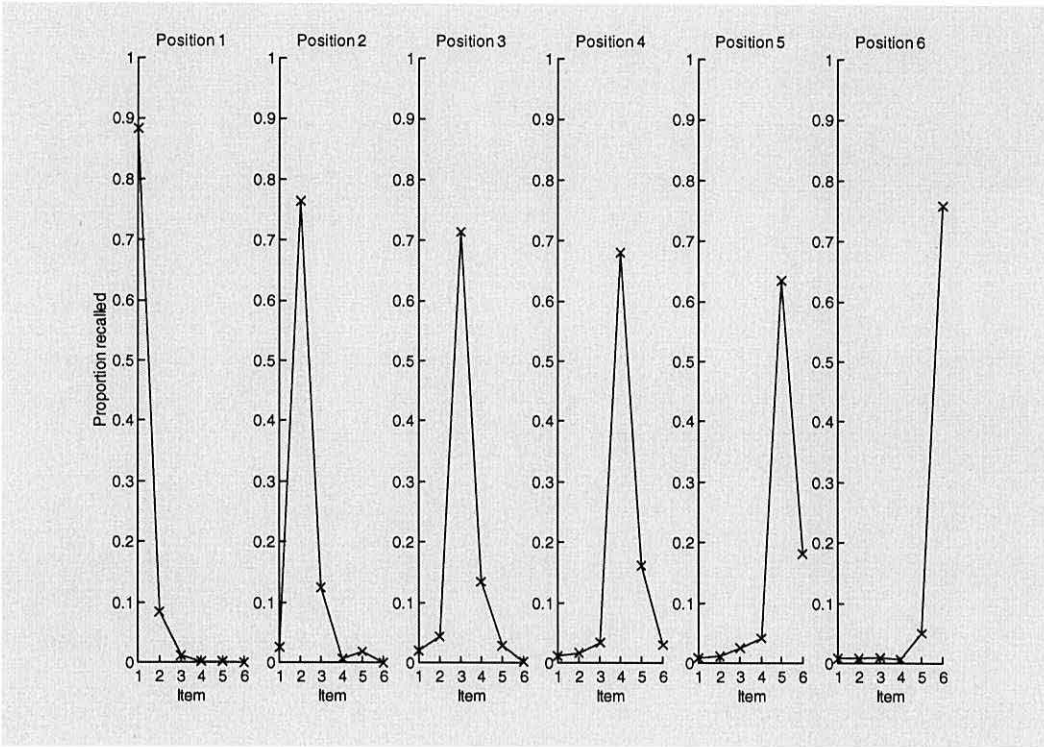

Figure 6.11 *Distance functions illustrating the effect of an inhibitory process preventing repeated item errors with the addition of a vocabulary of distractor items during recall*

Unlike the previous simulation, if an item is unable to be retrieved correctly at its target serial position as it has already been recalled erroneously in one of the preceding list positions, OSCAR is not forced to select one of the remaining list items and commit an order error. Instead, OSCAR may select one of the distractor items and although the accuracy measure suffers for this serial position, the error does not affect each of the subsequent serial positions. For example, the first three items of a list may be recalled as *ABD*. In this case, the third item has been replaced by the target item for the fourth serial position. Its recall in the correct serial position is prevented by the inhibitory mechanism. Although it is possible that the item recalled for the fourth serial position is the letter *C* (and this would produce a neighbour transposition effect), here we assume that this is not the case. In the previous simulation, OSCAR would be forced to select one of the remaining vocabulary items presented during training: most probably *E*, which resulted in the increase in number of intra-list intrusions occurring towards the end of the list. However, in the present

case, OSCAR may retrieve one of the distractor items instead. Therefore, recall of the list may read *ABDX*, which benefits the latter list items by allowing the letter *E* to be recalled in its correct serial position: *ABDXEF*.

In this manner, it is possible to see how a vocabulary of distractor items can help improve recall of items towards the end of the lists *if* there is some inhibitory mechanism preventing recall of items recalled erroneously in prior serial positions. However, an alternate suggestion to reduce the number of forced intra-list errors towards the end of the list is to include a recall threshold, based upon the similarity between the retrieved item and each of the vocabulary items, below which items are omitted. This hypothesis is examined in section 6.7.

6.4.5 Summary

In this section, the effect of a simple inhibitory process on OSCAR has been investigated. Simulation 15 revealed that a very simple inhibitory process preventing the same item from being recalled in successive serial positions could have a marked affect on performance: reducing performance overall whilst introducing extended primacy and last item recency. Simulation 16 illustrated that a more complex inhibitory process, in this case one which prevented items being recalled twice in the same list regardless of serial position, could have a dramatic effect on serial order performance: eliminating the last item recency effect by forcing many more intra-list errors in the last serial position. However, by introducing a vocabulary of distractor items, as reported in simulation 17, this recency component may be returned to the serial position curve and a more natural order error distribution produced.

6.5 Adding noise during learning or recall

6.5.1 Introduction

Up to this point, the only degenerative aspects of the model's behaviour have been due to the distinctiveness of the context vectors, the limited capacity of the Hebbian memory matrix, the item-to-context associations decaying with time in the weights

matrix or the effect of varying the strength with which these associations are stored in the memory.

For the following simulations (18, 19 and 20) interference is added to the network by adding noise to the Hebbian matrix in the "intervals[12]" between each association being learned or recalled. Output interference is produced as a result of a small amount of random noise being added to each element of the memory matrix immediately after an item has been recalled from the lexicon. Input interference is produced by adding small amounts of random noise to each element of the memory matrix immediately after each new association has been stored in the memory trace. The "learning noise" condition is motivated by the intuition that if learned items are only using every (e.g.) third or fourth context vector, there must also be some information associated with the "unused" contexts during learning. The "recall noise" condition is motivated by implementing interference based forgetting or the idea that each item is recoded in the memory trace after recall. Interference during recall is expected to improve primacy effects by degrading recall performance for later items.

6.5.2 Simulation 18: *Adding noise during learning*

Introduction

During the following simulation, noise is added during the learning stage, immediately after each item-to-context association is combined with the current contents of memory. More specifically, after each association is stored in the composite memory matrix, small amounts of random noise are added to each element of the matrix. In this manner, a degree of input interference besides that due to the similarity of the item-to-context associations is implemented in the model. As has already been stated, there is an upper limit on the capacity of the network for the

[12] These "intervals" are due to the manner in which context distinctiveness is modelled. During learning, every n^{th} context vector is associated with a list item. If the contexts are very distinct, then n will be large (e.g. four or five) whilst if they are less distinct, then n will be quite small (e.g. one or two). Clearly modelling distinctiveness in this manner means that there will be many "unused" context vectors that are not associated with stimuli items. Here we propose that some form of input interference during learning occurs as a result of the storage of the association between the cognitive state at these times and the unused contexts. We model these associations as random noise.

number of associations of non-orthogonal vectors that can be stored and retrieved successfully from a Hebbian matrix.

By adding noise to the memory matrix during learning, we are reducing its capacity and hence the ability to accurately store and recall item-context associations. We would predict either that items earlier in the sequence will suffer the most as they will be degraded more than later items by the addition of noise in the network, or that performance will be reduced for each serial position as the capacity of the network for storing associations breaks down.

Method

The same procedure as was employed for experiment 10 is used in the current simulation. The context distinctiveness was controlled by fixing the inter-context spacing in each condition to four. However, after each item-to-context association was stored in the composite memory matrix, a small amount of random noise was added to each of the matrix elements. The proportion of learning noise was varied between 0%, 50% and 100%. Performance is recorded in terms of the serial position curve for each condition.

Results

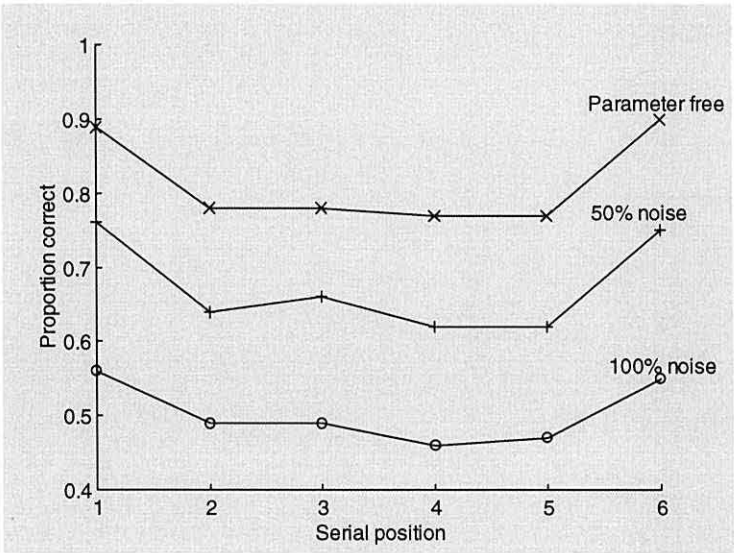

Figure 6.12 *Adding increasing levels of noise to the Hebbian memory matrix during learning*

The results of this simulation are presented in the serial position curve of figure 6.12. Clearly, adding noise during the "intervals" between each item-to-context association being learned degrades performance uniformly across serial position. The serial position curves in both the 50% and 100% learning noise conditions retain the symmetric bowing of the parameter free condition. However, mean performance drops from approximately 0.85 to 0.69 for the 50% condition and then approximately 0.5 for the 100% noise condition.

Discussion

Clearly we must reject our earlier prediction that adding noise to the network in this manner would degrade performance only in the early list positions. Instead, learning noise degrades performance uniformly across serial position. In retrospect, given the limited capacity of the Hebbian matrix for storing non-orthogonal vectors of continuous features, this is not surprising, as the memory matrix is effectively storing the information of six associations plus six "noisy associations". Next we consider the effect of adding noise during recall: output interference.

6.5.3 Simulation 19: *Adding noise during recall*

Introduction

During the following simulation, noise is added to the memory matrix during the item recall stage. After each item has been recalled from memory, a small amount of noise will be added to the contents of the memory trace. It is anticipated that, in contrast to the previous simulation, this will introduce primacy by degrading the later associations in the sequence.

Method

The same procedure as was employed for experiment 10 is used in the current simulation. The context distinctiveness was controlled by fixing the inter-context spacing in each condition to four. However, after each item-to-context association was recalled from the composite memory matrix, a small amount of random noise was added to each of the matrix elements. The proportion of noise was varied from

0% to 50% and then 100%. Performance in each condition is presented as the serial position curve for each condition.

Results

The results of this simulation are presented in figure 6.13. Clearly adding noise during recall reduces performance for all but the first item. Recall drops from 90% in the first serial position to a trough of 52% in the fourth. There is a small recency effect extending over the fifth and sixth items with performance in the last serial position approximately 58%.

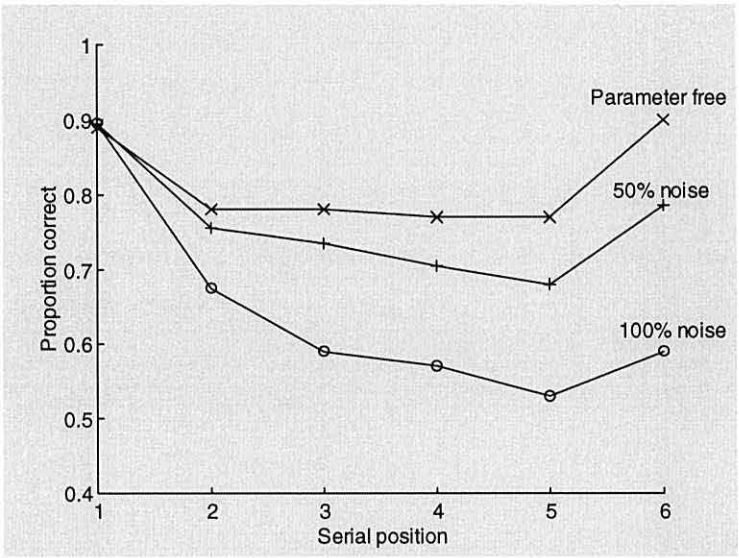

Figure 6.13 *Adding increasing levels of noise to the Hebbian matrix during recall*

Discussion

The results of this simulation illustrate that adding noise during recall, to introduce interference, clearly benefits the early list items and introduces a large primacy effect as a result. Performance for the first item is unaffected by the noise as a result of the method by which noise is added: i.e. after each item has been recalled. For subsequent items, the effect of noise is huge and is reflected in the 28% drop in performance in the second serial position. Next we examine the effect of combining learning and recall noise on the serial position curve.

6.5.4 Simulation 20: *Adding noise during learning and recall*

Introduction

During the following simulation, noise is added to the Hebbian matrix both during the learning stage after each association has been stored in the memory matrix, and during the recall stage after each item has been retrieved from memory. In light of the results of the previous two simulations, it is predicted that adding noise in both these intervals should reduce performance over each serial position while maintaining the extended primacy effect and last item recency revealed by the previous simulation.

Method

The same procedure as was employed for experiment 10 is used in the current simulation. However, after each item-to-context association was stored in the composite memory matrix, a small amount of random noise was added to each of the matrix elements. Then, after each item-to-context association was recalled from the composite memory matrix, a further amount of random noise was added to each of the matrix elements. The proportion of noise (both during learning and recall) was constant at 50%. Performance in each condition was recorded as a serial position curve. The context distinctiveness was controlled by fixing the inter-context spacing in each condition to four.

Results

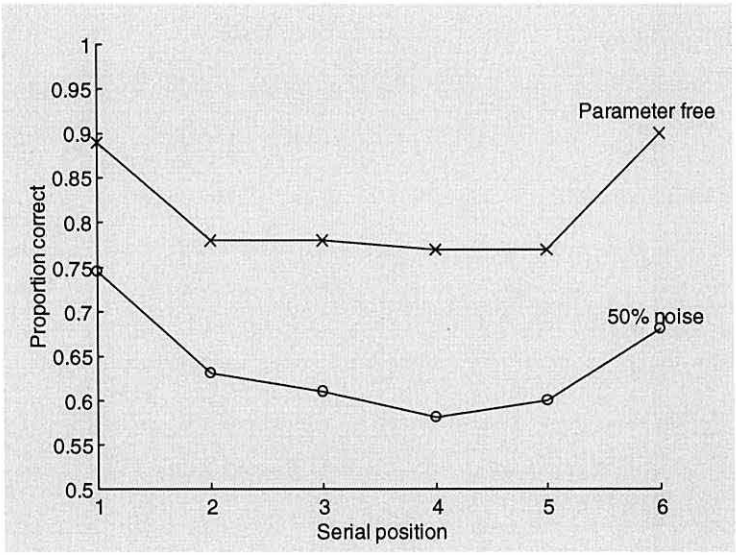

Figure 6.14 *Adding noise to the Hebbian matrix during both training and recall*

The results of this simulation are presented in the serial position curve of figure 6.14. This illustrates how performance in the noise condition in each of the six serial positions is poorer than in the corresponding positions in the parameter free condition. As predicted, there is also an extended primacy effect over the first three serial positions (75% in the first position, then 63% and 61%) before reaching the trough of 58% in the fourth serial position. Recency extends over the last two serial positions, with last item recency of 8%.

Further analysis of the effect of noise in learning and recall is provided by the distance function of figure 6.15 which reveals that order errors occur uniformly in each of the serial positions *at least* two positions away from the target serial position. For example, the fourth, fifth and sixth items are each recalled in similar proportions in the second serial position. Similarly, the first, second and third items in the sixth serial position.

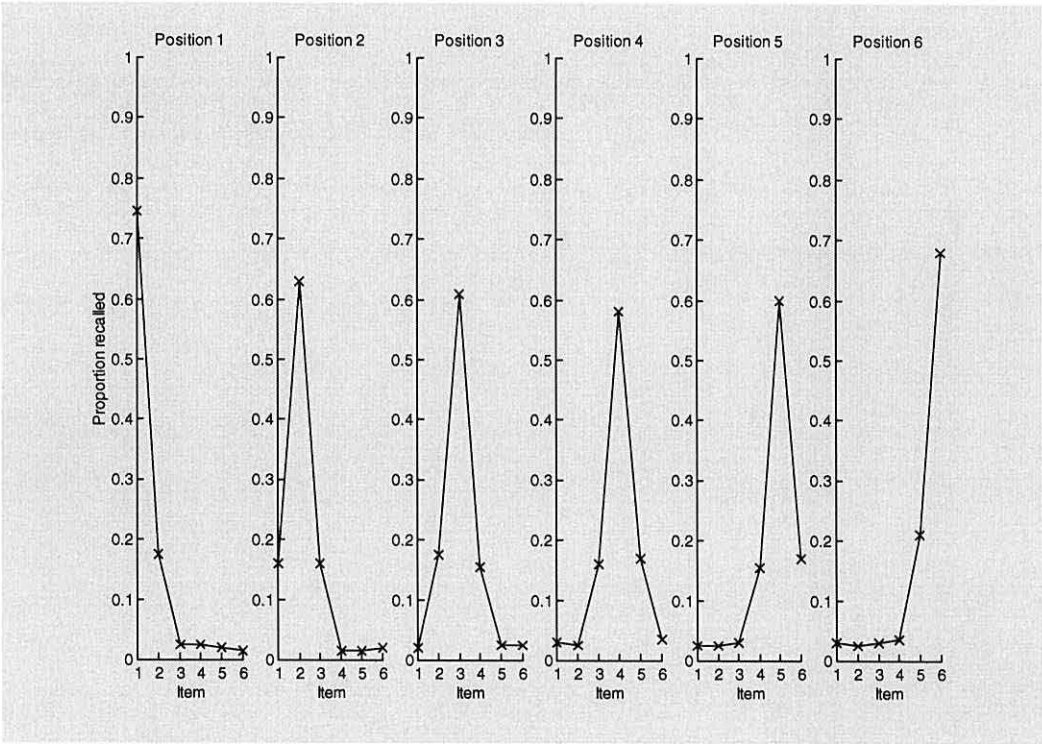

Figure 6.15 *Distance functions illustrating the effect of adding noise to the Hebbian matrix during both training and recall*

Discussion

These results (figure 6.14) illustrate that adding noise during both learning and recall degenerates performance overall (due to the impact of learning noise on the Hebbian matrix's capacity for associations) whilst introducing extended primacy (due to the output interference provided by the noise during recall).

The distance function of figure 6.15 contrasts with the distance function of the parameter free condition (figure 5.16) where items are only recalled erroneously in the serial positions immediately adjacent to the target position. In the present case, order errors occur evenly across all serial positions for all items in addition to those order errors that occur about the target serial positions in accordance with Estes (1972).

6.5.5 Summary

Clearly adding noise to the model can have a significant impact on performance for serial ordered recall. If noise is added to the memory trace during the "intervals" between each association being stored in the composite memory matrix, *learning noise*, performance is reduced evenly across each serial position (figure 6.12). Conversely, if *recall* noise is added to the memory trace during the "intervals" after each item has been recalled from the composite memory, performance is reduced for the later items and extended primacy introduced. If components of both learning and recall noise are added to the network, performance overall is reduced while maintaining the extended primacy and last item recency desired of the model.

Introducing output interference by adding noise during recall will clearly reduce the recency exhibited by the items occupying the latter half of the list positions. Furthermore, introducing small amounts of input interference will reduce performance across all list positions. However, before OSCAR can be applied to the problems of fitting a range of empirical benchmarks, the effect of item confusability must first be addressed.

6.6 Simulation 21: *Effects of item similarity*

Introduction

All the data presented thus far has been generated using nonconfusable or dissimilar item vectors. Each vector element is selected randomly from a normalised distribution about zero with variance of one before the vector is normalised to improve the recall deblurring process. However, as much of the empirical data in chapter 2 considered the effects of using phonologically similar stimuli (e.g. Conrad, 1964; Conrad & Hull, 1964; Wickelgren, 1965a; Baddeley, 1968; Henson, Norris, Page & Baddeley, 1996), it is necessary to consider the effect that making a proportion of each item vector identical will have on the model. It is unclear exactly how best to model phonemic similarity given the nature of stimuli in computational models of memory (Eich, 1982; Lewandowsky & Murdock, 1989). However, in accordance with a number of previous models, similarity is implemented by employing a degree of overlap between one stimulus item and another. More precisely, where stimuli are represented as n-dimensional vectors of features, similar or confusable stimuli share a proportion of features, or elements, in common.

As has already been illustrated (Chapter 4; Brown, Preece & Hulme, 1995) it is important that item normalisation remain intact when item confusability is introduced. To summarise the method described in the appendix, item vector stimuli are generated by filling each vector element with a scalar randomly selected from a normalised distribution. The first item is used as the standard and is normalised first. Next, before the remaining *confusable* stimuli are normalised, a percentage (typically 25%, 50% or 75%) of the normalised elements of the first item are copied to each of the remaining item vectors. Then a process of normalisation is undertaken which leaves those elements copied from the first item undisturbed, while ensuring that we have a set of confusable yet normalised item vectors.

In this simulation, the effect of increasing the confusability of the stimuli is presented. The predicted results are twofold: firstly, that performance, measured in terms of the serial position curve, will reduce as the degree of confusability increases;

secondly, that this reduction in overall performance will be reflected in an increase in the proportion of order errors that occur (Wickelgren, 1965a).

Method

The same procedure as was employed for experiment 10 is used in the current simulation. However, items selected from four different vocabularies of stimuli were presented during each condition. In the first condition, nonconfusable stimuli were selected. In each of the remaining conditions, increasingly confusable stimuli were presented: each with either 25%, 50% or 75% overlap. For each of the four conditions, a distinct set of contexts was used, generated with an inter-context spacing of four simulated time steps.

Results

The serial position curves generated for each of the nonconfusable and then increasingly confusable conditions are illustrated in figure 6.16. These illustrate clearly how increasing the inter-item confusability results in an even decrease in performance across each serial position.

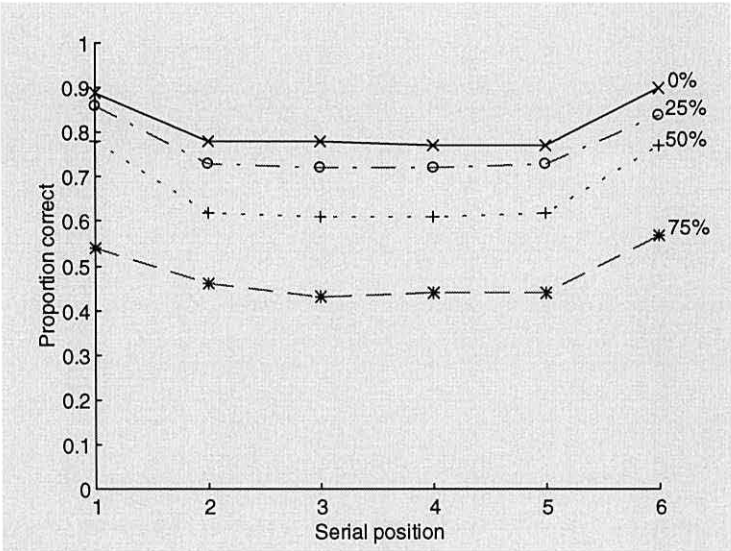

Figure 6.16 *The effect of increasing item similarity (% overlap) on serial order performance*

Furthermore, analysis of the proportion of order errors by presenting the distribution of item recalls in the third serial position for each condition (figure 6.17) reveals that as the stimuli become more confusable, so order errors increase. This is reflected in

the manner in which the spike about the target serial position (cf. the nonconfusable condition) broadens as stimulus confusability increases (cf. the 75% confusable condition). For example, the first, fifth and sixth items are recalled infrequently in the third serial position in the nonconfusable condition. However, in the 75% confusable condition, these same items are recalled in approximately 7% of trials. This (21%) increase in intra-list errors is reflected in the reduction of correct recalls.

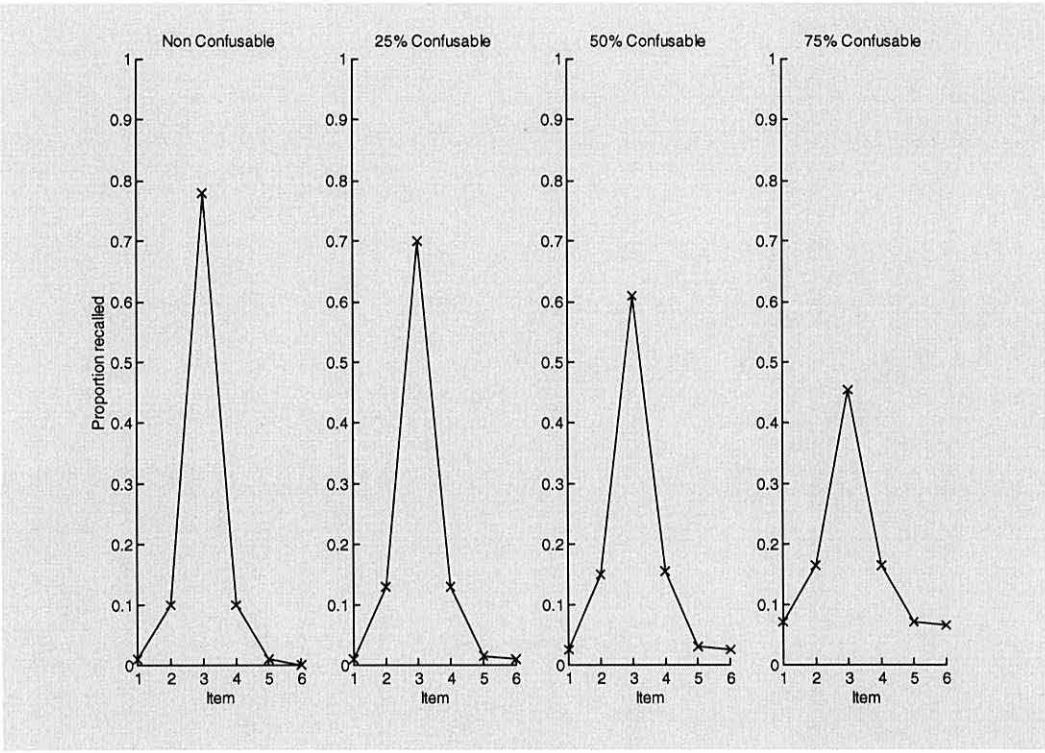

Figure 6.17 *The effect of increasing item similarity on distribution of order errors in the third serial position*

Discussion

It is evident from these results that performance for the parameter free version of OSCAR is reduced dramatically as the similarity between each stimuli item increases. As would be expected, the increase in item similarity has little effect on the shape of the serial position curve: there are still reasonable recency and primacy edge effects. Analysis of the distance function in the third serial position for each condition reveals that as similarity increases, so order errors occur in greater proportions as distant items are recalled more often in the third serial position.

However, results so far have concentrated on the distribution of order errors. Although OSCAR is capable of producing item errors when the lexicon of recallable items contains unlearned, distractor, items, it is not capable of omitting to recall an item. In the next section, this deficiency is addressed by the addition of a mechanism which introduces item omission errors into the OSCAR architecture.

6.7 Simulation 22: *Omission errors*

Introduction

There remains a highly significant parameter that needs to be described before simulations of the empirical data can be attempted. As the model stands at present, if an error occurs during recall and it is unclear which item should be retrieved from the lexicon of recallable items, OSCAR will be forced to recall the item that is most similar to that retrieved from the memory trace. However, this means that OSCAR will always recall something for every serial position, with the result that errors may be forced to occur in the later serial positions (e.g. simulation 16). However, we know from the empirical data (Conrad, 1965; Henson, Norris, Page & Baddeley, 1996) that subjects often fail to recall *any* item. Conrad (1965) suggests that this may be due a failure in recognising the signal that corresponds to the learned item above the background noise. Clearly, with the addition of output interference such as that described in simulation 19, there is a high probability that the associations will become lost in the noise of the Hebbian matrix.

Omission errors may be introduced to the model by the addition of a noise threshold below which items are omitted during recall. More precisely, when the retrieved item is compared with each of the recallable lexicon items, if the similarity of retrieved item to each recallable item falls below an arbitrary threshold, then an omission is recorded and recall proceeds to the next serial position (cf. Conrad, 1965; Page and Norris, 1995).

In this manner it is possible to introduce omission errors into the model. In the current simulation, the distribution of omission errors is considered.

Method

The procedure employed during the following simulation was identical to that described in experiment 10 but for the following differences. The context vector distinctiveness was controlled by setting the inter-context spacing to four. The decay rate parameter described previously was set to a value of 0.9. A non-linear learning rate parameter was introduced and took a value of unity for the first association and then 90% of the previous value for each of the subsequent associations (cf. figure 5.17). Furthermore, an arbitrary noise threshold was set in order to introduce omission errors. During recall, if the value of the cosine between the retrieved item and the lexicon item fell below the threshold (noise threshold = 0.5), recall of that item was prevented. If every available lexicon item failed to be recalled, an omission was recorded. Results are presented as the proportion of errors which were omission errors for each serial position.

Results

The results clearly demonstrate that omission errors account for increasing proportions of all the errors across serial position. In the first position, approximately 31% of errors are omissions. This increases to approximately 53% in the third, fourth and fifth serial positions, before peaking at 85% in the sixth serial position.

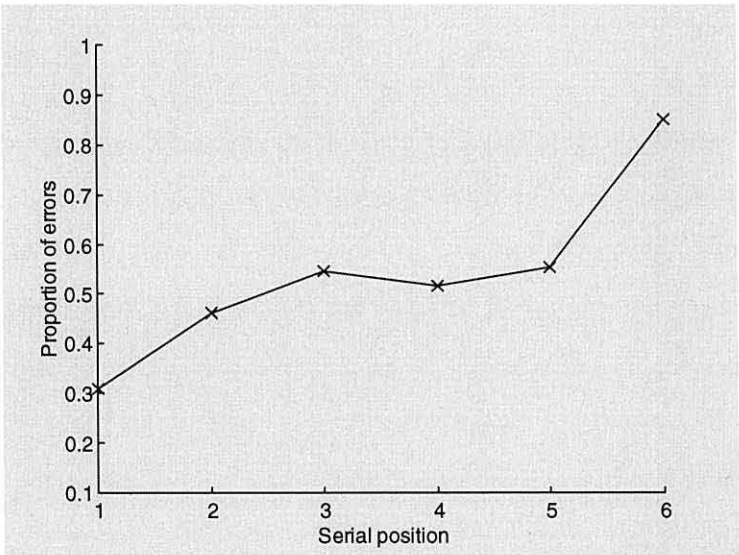

Figure 6.18 *Proportion of errors that are omissions for each serial position*

Discussion

Clearly the results presented here are in accordance with the empirical data which demonstrates that when omission errors occur, they do so most often in the last serial position (cf. Burgess & Hitch, 1992, table 3).

6.8 Summary

This chapter has explored the range of free parameters introduced to the OSCAR architecture. Here we summarise each parameter briefly while in the following chapter, the model is applied to the problem of reproducing a range of empirical data.

Context vector distinctiveness

This parameter is controlled by manipulating the inter-context spacing and influences the overall performance of the model by controlling the distinctiveness, or confusability, of the learned contexts. The use of "highly distinct" contexts produces a high level of performance. "Less distinct" contexts introduce errors due to the similarity of the reinstated contexts resulting in poorer performance overall. In any one simulation, the context distinctiveness is fixed and constant (e.g. figure 6.1).

Decay

Forgetting is implemented by introducing a weights decay parameter that scales the contents of the current memory trace prior to storing the next association. The decay parameter introduces recency to the model (e.g. figure 6.3) and is constant for any one simulation. Note that the weights decay during both learning and recall.

Learning rate

The non-linear learning rate reflects the intuition that list items become progressively less surprising or attention-demanding as a sequence advances. The learning rate introduces a primacy effect into the serial position curve (e.g. figure 6.4) and used alongside weights decay is sufficient to produce a reasonable serial position curve (e.g. figure 6.5, condition 3).

Output inhibition

Two different types of output inhibition have been explored. "Simple inhibition" prevents any one item being recalled in two successive serial positions (e.g. figure 6.6). A more complex inhibitive process that provides "repeated item inhibition" prevents the same item being recalled at *any* two serial positions (e.g. figure 6.8). This latter process eliminates recency from the serial position curve unless a larger lexicon of recallable items is provided.

Noise

If small amounts of noise are added to the weights matrix during learning, performance is reduced uniformly across each serial position (e.g. figure 6.12). This reflects the limited capacity of the Hebbian matrix for storing associations of non-orthogonal vectors. If output interference, another source of forgetting, is implemented by adding small amounts of noise to the weights matrix during recall, recency becomes eliminated from the serial position curve (e.g. figure 6.13). However, if small amounts of both types of noise are added, it is possible to produce a serial position curve with extended primacy and last item recency (e.g. figure 6.14).

Item confusability

As items are represented by vectors of features, it is possible to introduce inter-item confusability by overlapping stimuli i.e. letting different stimuli items share features in common. Increasing the confusability of the stimuli in this manner reduces performance overall (e.g. figure 6.16) and introduces order errors (e.g. figure 6.17).

Noise threshold

If a noise threshold is introduced to the model during the retrieval stage, below which items are not recalled, it is possible to introduce omission errors to supplement the order errors and few item errors that occur naturally in OSCAR.

Conclusion

Having gained a general computational understanding of OSCAR's behaviour, we are now in a position to apply the model to specialised psychological data. This is the task of the next chapter.

CHAPTER 7

OSCAR simulations of empirical data

7.1 Introduction

In the following section, the OSCAR model is applied to the problem of modelling the empirical data reviewed in chapter 2 and summarised in table 7.1.

OSCAR is first shown to reproduce the serial position curve (Baddeley, 1968) before being applied to the problem of modelling memory span (Crannell & Parrish, 1957). The effect of acoustic similarity on memory span is addressed (Conrad, 1965; Baddeley, 1966). In simulation 26, the effect of increasing the size of the item set is examined (Drewnowski, 1980). In the first of three related simulations examining the distribution of order errors, OSCAR attempts to reproduce the transposition gradient data of Henson, Norris, Page and Baddeley (1996). Next, we consider OSCAR's ability to replicate the characteristic "sawtooth" serial position curves of Baddeley's (1968, experiment 5) phonemic similarity effect. Simulation 29 examines the distribution of item and order errors further and compares them with the distance functions of Healy (1974).

Table 7.1

Summary of empirical data addressed by OSCAR

Simulation	Empirical result	Source of data
23	Serial position curve	Baddeley, 1968
24	Memory span	Crannell and Parrish, 1957
25	Acoustic effect on memory span	Baddeley, 1966a
26	Size of item set	Drewnowski, 1980
27	Transposition gradient	Henson et al., 1996
28	Phonemic similarity effect	Baddeley, 1968
29	Item and order errors	Healy, 1974
30	Partial reinstatement of context	Murdock, 1968

The final simulation in this chapter examines the partial report serial position curves of Murdock (1968) to illustrate the effect of assuming that the dynamic-learning context signal can be only partially reinstated at the time of retrieval.

7.2 Simulation 23: *Serial position curve*

Introduction

One basic result from serial order memory experiments is the serial position curve. The data we are concerned with is obtained from subjects who are presented visually with a list of items and required to recall them, immediately or after a short interval, in the order in which they were presented. Performance is measured by reporting the probability that an item will be recalled correctly in its correct list position.

Example serial position curves were illustrated in chapter 2 (e.g. Jahnke, 1963; Baddeley, 1968; Murdock, 1968). However, although the precise nature of the curve depends upon the details of the recall task, the data have a number of properties in common.

The most significant properties of the asymmetric serial position curve include the extended *primacy* effect over the first few serial positions and the last item *recency*. The primacy effect is the high performance exhibited during recall for the items occupying the earliest list positions. For visually presented stimuli, this extends over a number of items. The recency effect is a similar property but for the latter half of the list. Typically recency extends over two or three items. However it may only appear for the last item in a visually presented list. These elements are clearly visible in the Baddeley (1968) data illustrated in figure 2.7 where letters were presented visually to the subjects.

In the following simulation, OSCAR is applied to the basic properties of the serial position curve. Specifically, we attempt to fit OSCAR to the Baddeley (1968, experiment 5), *DDDDDD* data. This data corresponds to the six nonconfusable (i.e. six different, *D*) letters condition and has been selected as the stimuli are presented

visually and recall is immediate. (We will return to these data in simulation 28 where OSCAR is applied to the problem of phonemic confusability in alternating list conditions).

Method

For the present simulation, OSCAR was presented with lists of six previously unseen items. Items of dimensionality 16 with elements drawn randomly from a normal distribution about zero and variance of one were normalised. During training, each item was associated with a 16 element context vector by Hebbian learning and the association stored in the composite memory matrix. Recall involved reinstating each context vector from the sequence in order and presenting it as a probe for recall to the memory store in order to generate a retrieved item for each serial position. This process was repeated for each of the 20 sets of context vectors, each using the same set of vocabulary items, and the retrieved items averaged accordingly. Finally, each of the retrieved items was compared with each of the items in the lexicon of recallable items. The lexicon item that was most similar to the retrieved item was recalled as the output for that serial position. In order to average the results, a new vocabulary of items was generated and the training and recall process repeated with each set of context vectors.

OSCAR's free parameters were assigned values in order to maximise the quality of the fit to the empirical data. The distinctiveness of the dynamic learned-contexts was controlled by setting the inter-context spacing to five. Forgetting was provided by the weights decay[13] parameter (0.90) and output interference (20% noise). Performance was reduced overall by introducing a small amount of noise during learning (20% noise). The noise threshold below which items are omitted was set to zero. The non-linear learning rate ($\lambda_{i+1}=0.9\lambda_i$) ensured that the model exhibited primacy. Finally, a lexicon of recallable items consisting of the six items presented during learning and an additional six items was present during the retrieval and recall process. An inhibitory process prevented repeated item errors. The results were averaged over 2000 trials.

[13] Note that the weights continued to decay during recall.

The mean proportion of items recalled correctly in each serial position was recorded and the results presented as a serial position curve.

Results

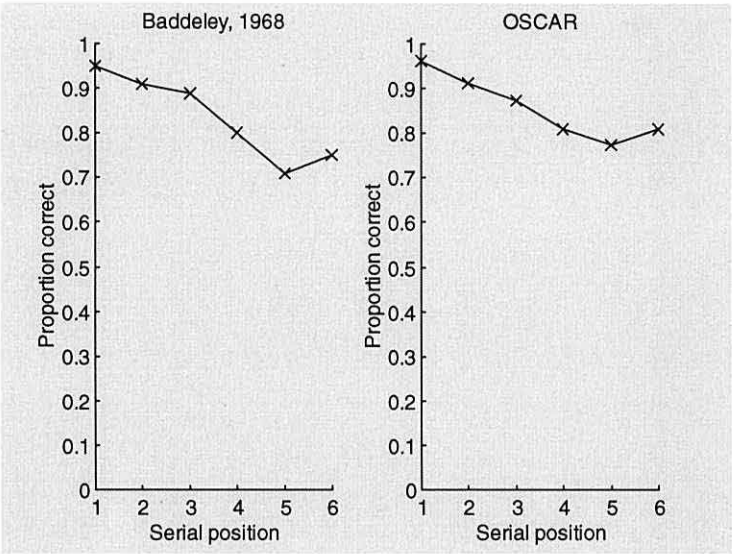

Figure 7.1 *OSCAR modelling the serial position curve of Baddeley (1968, experiment 5, DDDDDD condition)*

The serial position curve generated by OSCAR in this recall task is presented alongside the target empirical result of Baddeley (1968) in figure 7.1. OSCAR exhibits an extended primacy effect with recall performance decreasing steadily over the first four serial positions from 96% to a low of 77% in the fifth serial position (cf. Baddeley's data which drops from 95% to 71% in the fifth serial position). There is a last item recency effect of 4%, identical to Baddeley's data.

Discussion

It is clear from figure 7.1 that OSCAR possesses the correct attributes for a serial position curve: there is an extended primacy portion of the curve and there is a degree of last item recency, although it is not an exact fit of the Baddeley data as performance for the latter half of the list is too high. However, there is some variation in the form of the serial position curve, and given that OSCAR exhibits the main properties of a serial position curve, it would be fair to say that OSCAR has provided an adequate fit to the empirical data.

In fact, the parameters selected for this simulation were not selected in order to minimise the error between the serial position curves of Baddeley and OSCAR. Instead, they were selected in order to capture Baddeley's (1968, experiment 5) phonemic similarity effect described in Simulation 28 and as such provide a better fit to the data overall than to the nonconfusable stimuli condition specifically.

The primary features of the model that contribute to the observed serial position curve are the components that introduce primacy. Most notably, these are the reducing learning rate and the interference due to noise during recall. The components responsible for recency include the weights decay. However, as was illustrated by the simulations of the previous chapter, the inhibitory mechanism coupled with the extended range of recallable lexicon items may also be partly responsible for the recency effect during recall.

7.3 Simulation 24: *Memory span*

Introduction

In a memory span simulation, subjects are required to recall all of the items learned in a single trial correctly and in the correct order. When this probability of correct responses is recorded for different list lengths, a characteristic 'reverse-S' shaped curves (figure 2.1) is produced. Performance, which is initially 100% for small lists, decreases as the list length increases until it drops to 0% for lists of 10 or 11 items or greater. Memory span is defined as being the number of items that are correctly recalled 50% of the time. Typically this is approximately seven items. The precise nature of the memory span curve depends on the nature of the stimuli: Crannell and Parrish (1957) demonstrate that performance is better for digits than words. In order to model the effect of using different types of stimuli, OSCAR's learning rate parameter is varied to reflect the assumption that the better-remembered items will be learned more strongly (cf. Lewandowsky & Murdock, 1989).

Method

This simulation employed the same procedure and parameter values as were used in simulation 23 with the following variations. Forgetting is provided by the weights decay parameter (0.87) and output interference (10% noise). Performance is reduced overall by introducing a small amount of noise during learning (10% noise). The precise number of stimuli items, the *list length*, was varied between two and 11 items. The non-linear learning rate was varied, taking values of 0.9, 0.8 and 0.7 to reflect the different types of stimuli. The learning rate of 0.9 corresponds to digit stimuli, of 0.8 to letter stimuli and 0.7 to word stimuli. No distractor items were added to the lexicon of recallable items and no inhibitory processes were used. The mean proportion of completely correct sequences recalled, memory span, was recorded for each list length.

Results

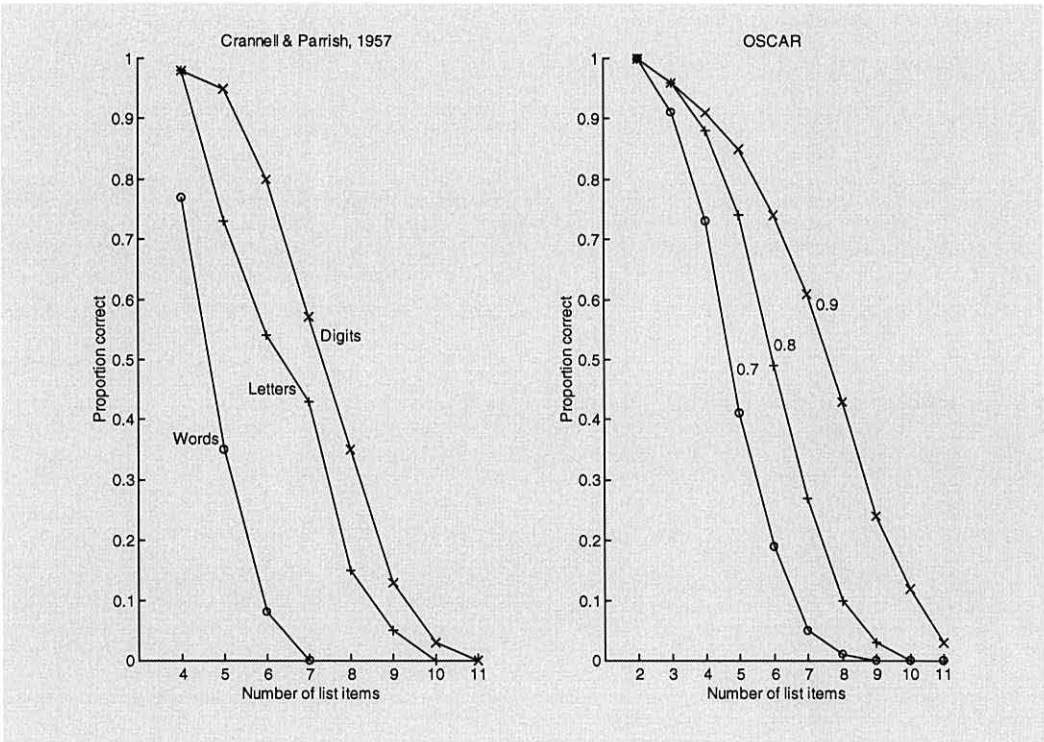

Figure 7.2 *OSCAR memory span for digits (λ=0.9), letters (λ=0.8) and words (λ=0.7) compared with Crannell & Parrish (1957)*

The results of this simulation are presented in figure 7.2 alongside the empirical data of Crannell and Parrish (1957). For the digit condition (λ=0.9), recall decreases from 100% for two item lists to 0% for eleven item lists. The memory span for OSCAR in this condition, measured as the number of items recalled accurately 50% of the time, is 7.1 items. Memory span for letters (λ=0.8) is approximately 6 items and for words (λ=0.7), memory span drops to approximately five items.

Discussion

These results demonstrate that OSCAR is capable of reproducing the form of the empirical data for memory span of Crannell and Parrish (1957). Manipulating the learning rate parameter to model the effects of learning different stimuli provides an adequate fit to the digits, letters unlimited and words unlimited conditions of Crannell and Parrish (1957). Significantly, the memory span of 7.1 for digits is in line with empirical findings (e.g. Crannell & Parrish, 1957; Miller, 1956).

Although similar parameters to those responsible for the serial position curve of the previous simulation are also responsible for the nature of the memory span functions, these curves also reflect the capacity of the Hebbian memory matrix and would, therefore, also be susceptible to interference due to noise during learning.

7.4 Simulation 25: *Acoustic similarity and memory span*

Introduction

In chapter 2 we reviewed much evidence to suggest that performance for stimuli that sound alike is poorer than if the stimuli are distinct (e.g. Conrad, 1964; Conrad & Hull, 1964; Wickelgren, 1965a, 1965b). In a series of simulations, Baddeley (1966) investigated the affect of acoustic, semantic and formal similarity on memory span for lists of five words. Baddeley reported that recall was significantly better for lists of acoustically nonconfusable words (9.6%) than for acoustically confusable words (82.1%). In the following simulation, we aim to replicate these specific findings using OSCAR. In the simulation, item similarity is modelled as overlapping features.

Method

This simulation employed the same procedure and parameter values as were used in experiment 23 but with the following variations. Context distinctiveness was controlled by setting the inter-context spacing to four simulated time steps. Forgetting was provided by the weights decay parameter (0.87) and output interference (10% noise). Performance was reduced overall by introducing a small amount of noise during learning (10% noise). The confusable items overlapped 75% of features. No distractors items were added to the lexicon of recallable items, nor were any inhibitory processes used during recall. The mean proportion of completely correct sequences recalled was recorded for both dissimilar (nonconfusable, *NC*) and similar (confusable, *C*) items.

Results

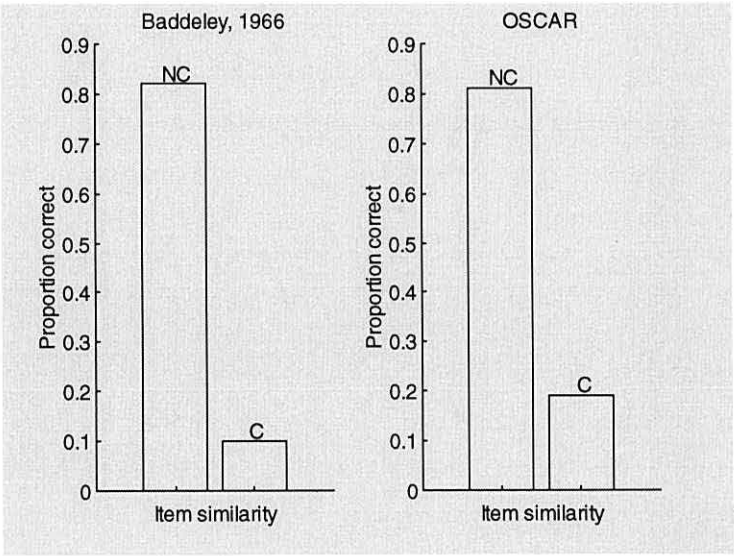

Figure 7.3 *OSCAR fit to effect of acoustic similarity on memory span (Baddeley, 1966)*

The results of this simulation are presented in figure 7.3 alongside a reproduction of Baddeley's (1966a) results for verbally presented acoustically confusable and nonconfusable items. Performance for nonconfusable items (81%) is comparable with that reported by Baddeley (82.1%), although there is greater discrepancy between the model and the empirical data for the confusable items condition (19% recalled by OSCAR versus 9.6%).

Discussion

These results confirm that confusable stimuli are harder for the model to recall accurately. Previously we demonstrated how serial position curve performance suffered as item confusability increased (simulation 21, figure 6.16). Here we demonstrate that the same effect can be seen for memory span and that the magnitude of the effect in OSCAR is comparable to that exhibited by the empirical data. Simulation 21 also illustrated that this reduction in performance reflects an increase in order errors. Although this is not explored here, in simulation 27 distance functions for both nonconfusable and confusable items will confirm this increase.

However, performance for the confusable items condition is almost twice that of the empirical data. This may be attributed to the arbitrary manner in which item similarity is implemented by models using distributed vector representations of stimuli. In the present case, given the accuracy of the associative mechanism, the similarity between the items would have to be increased beyond 75% in order to reduce performance further.

7.5 Simulation 26: *Vocabulary size effects*

Introduction

The following simulation examines the effect of manipulating the size of the lexicon of recallable items available to the model during the retrieval stage. This is significant as a similar mechanism is responsible for controlling the degree of recency exhibited by TODAM (Lewandowsky & Murdock, 1989; Mewhort, Popham & James, 1994; Nairne & Neath, 1994). However, Drewnowski (1980) reports that increasing the size of the vocabulary of recallable items should have no effect on memory for serial order for familiar items. Conrad and Hull (1964) also demonstrated that the size of vocabulary of recallable items made little difference to recall performance. Clearly this poses a problem for TODAM, so for the following simulation, the effect of increasing the lexicon of recallable items with OSCAR is addressed.

Method

This simulation employed the same procedure and parameter values as were used in experiment 23 but with the following variations. The size of the vocabulary of distractor items was varied between zero (i.e. there were no distractor items) and three (i.e. a total of the three times the number of learned items were added to the vocabulary during recall e.g. for a four item list, the vocabulary consisted of the four items plus a further 12 distractor items). The simulation was repeated for four different list lengths: four, five, six and seven item conditions. No inhibitory mechanisms were used during recall. Performance was measured in terms of the mean proportion of items recalled as a function of the number of items in the distractor vocabulary.

Results

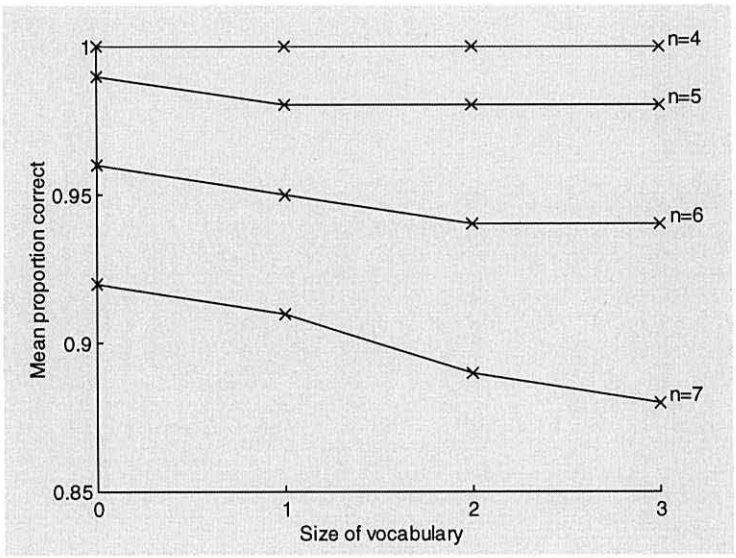

Figure 7.4 *Mean proportion of items recalled correctly as a function of size of distractor vocabulary (list length, n=4,5,6 & 7)*

The results presented in figure 7.4 illustrate that increasing the number of distractor items available at recall has little or no effect on the mean proportion of items recalled correctly for each of the four list lengths. For the shortest list length (n=4), recall is unaffected at 100% for each of the conditions (i.e. where the vocabulary size is equal to four items plus either zero, four, eight or twelve items). A similar result is found in the five and six item conditions, where performance drops only 1% to 98% and 2% to 94% respectively. However, when there are seven items in the list, performance does

decrease slightly as the vocabulary increases: initially recall averages at 92%, but it drops steadily to 88% for the vocabulary of 21 items.

Discussion

Clearly these results are in accordance with Drewnowski (1980) as increasing the size of the lexicon of recallable items available during retrieval has little affect on performance. Any learned item that has not yet been recalled will always more likely to be recalled than an unlearned extra-list item. The chances of an extra-list item becoming activated when cued by a learned-context are very remote. Therefore, even when the vocabulary of recallable items is large, only very few intra-list intrusions should occur. However, if there are extra-list items that are similar to the learned items, we anticipate that more intra-list intrusions might occur. However, it is important to note that this data has been produced without the use of any of OSCAR's inhibitory mechanisms. Simulation 16 illustrated that when OSCAR employed an inhibitory mechanism preventing repeated items during recall, the recency component of the serial position curve would be eliminated unless a vocabulary of distractor items expanded the lexicon of recallable items (simulation 17, figure 6.10). A similar finding is reported with TODAM, where limiting the number of recallable items as a result of the sampling without replacement process controls the recency portion of the serial position curve (Nairne & Neath, 1994, figure 1).

7.6 Simulation 27: *Transposition gradients*

Introduction

Simulation 25 illustrated how memory span for confusable stimuli was poorer than that for nonconfusable items (figure 7.3; Baddeley, 1966). However, evidence reveals that this poorer performance is reflected in an increase in the number of order errors (Wickelgren, 1965a). Analysis of the distribution of order errors is possible if the proportion of items recalled in each serial position is presented. Distance functions reveal this distribution and should confirm that when OSCAR is presented with a list of confusable stimuli, many more order errors should occur than when the stimuli are nonconfusable. In the following simulation, OSCAR is compared with data presented

by Henson, Norris, Page and Baddeley (1996) in which transposition gradients for six item lists of both nonconfusable (figure 7.5) and confusable (figure 7.7) items are learned.

Method

This simulation employed the same procedure and parameters as were used for experiment 23 but with the following variations. The confusable items shared 50% of features in common. No inhibitory mechanisms were used during recall.

Results

The results of the simulation are presented in figures 7.6 and 7.8, for the lists of nonconfusable and confusable items respectively, alongside the corresponding figures from Henson, Norris, Page and Baddeley (1996).

Figure 7.6 reveals that, for nonconfusable items, OSCAR provides a good fit to Henson et al.'s data. The target item is recalled most often in each of the six serial positions. The majority of errors involve an item from one of the adjacent serial positions being recalled erroneously. Furthermore, there is extended primacy and last item recency of 8%. This is in line with the empirical data presented in figure 7.5 (Henson, Norris, Page & Baddeley, 1996, figure 3). For example, in the second serial position, the first item is recalled 10%, the second item 78% and the third item, 9%. The fourth, fifth and sixth items are not recalled at all in this list position. Henson et al. report for the same serial position that the first item is recalled 4%, the second item recalled 84%, the third item 6%. The fourth, fifth and sixth items are recalled approximately 2% of trials.

Figure 7.8 illustrates the results of exchanging the vocabulary of nonconfusable items with one of confusable items. Performance for the confusable condition is poorer than in the nonconfusable condition. As with the nonconfusable condition, the majority of the errors involve items from the serial positions immediately adjacent to the target serial position being recalled.

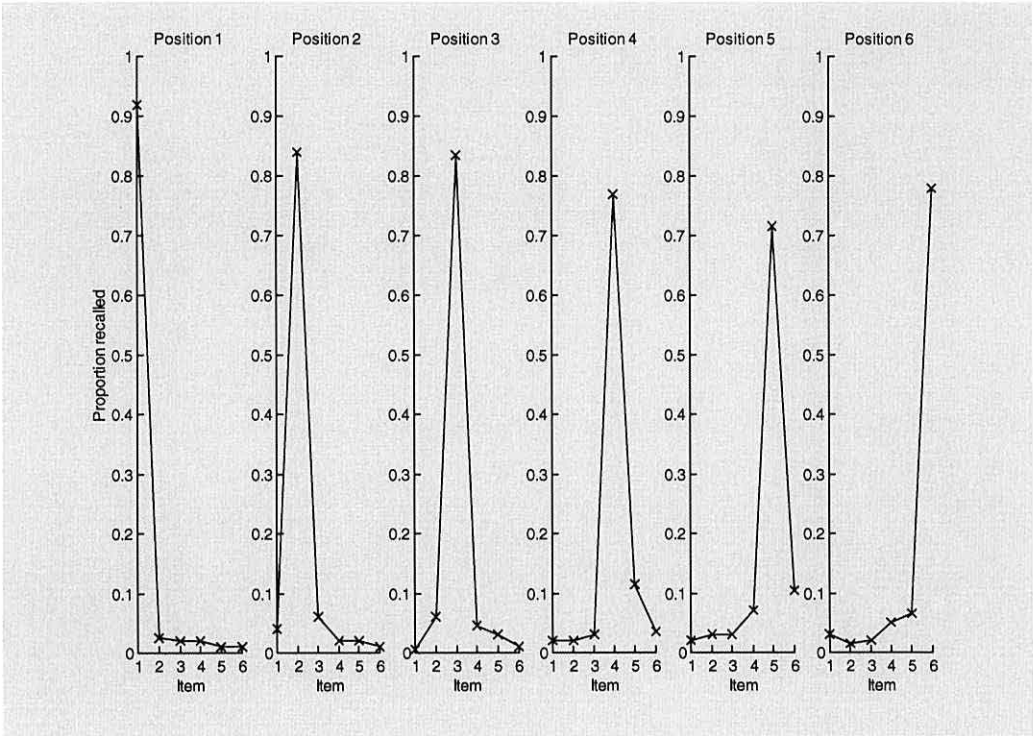

Figure 7.5 *Distance functions for six item lists of nonconfusable items (Adapted from Henson, Norris, Page & Baddeley, 1996)*

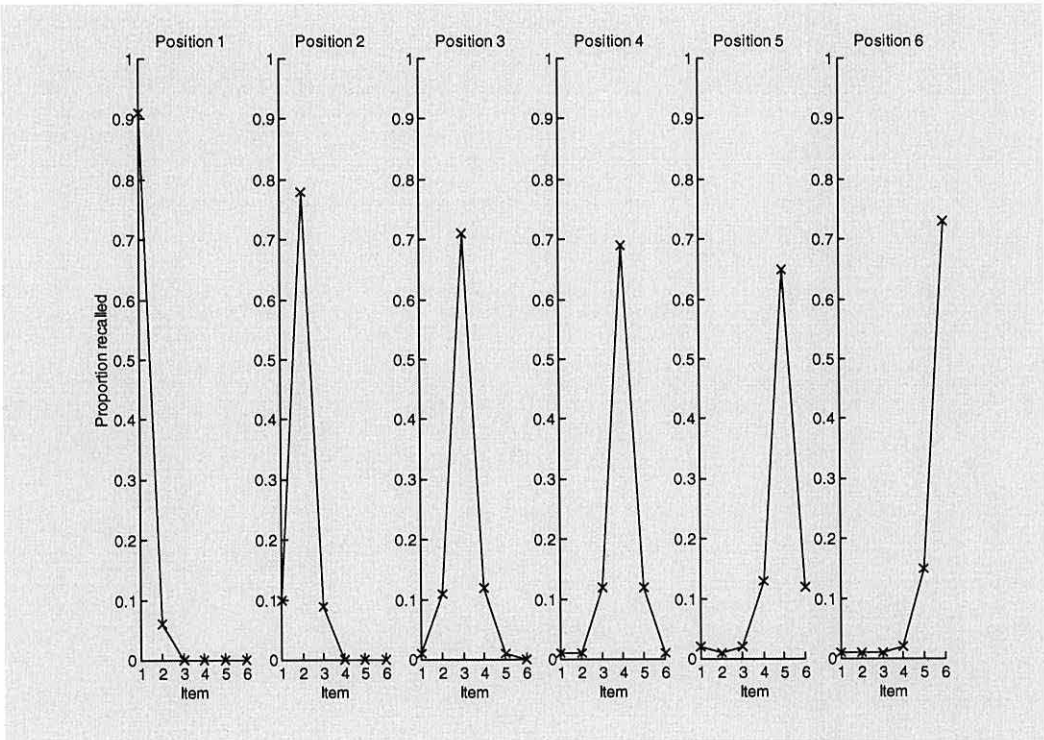

Figure 7.6 *OSCAR transposition matrices for six item lists of nonconfusable items*

Once again, the results compare favourably with the empirical data of figure 7.7 (Henson, Norris, Page & Baddeley, 1996, figure 2) with extended primacy and a last item recency effect of 8%. However, in contrast to figure 7.6, many more order errors involve items from more distant list positions. For example, in the sixth serial position, the first four items are each recalled approximately 4.5% of trials. However, it is also apparent that OSCAR commits fewer of these order errors than is evident from Henson et al.'s data. This may be accounted for by considering the artificial manner in which similarity is modelled in the OSCAR simulation.

Discussion

OSCAR would appear to have replicated Henson, Norris, Page and Baddeley's (1996) data with a reasonable degree of accuracy. Each transposition gradient reports maximal response in the target position. There is reliable primacy and recency in each of the conditions. Performance is considerably better for nonconfusable items than confusable. What is also very clear from these results is that, like the empirical findings, these error distributions are bounded by a *locality constraint* (Henson, Norris, Page & Baddeley, 1996), such that transposed items tend to be localised about their target serial position (Estes, 1972; Healy, 1974).

In the nonconfusable stimuli condition, OSCAR produces more order errors that consist of recalling the items immediately adjacent to the target item than is apparent in the empirical data. This effect may be attributed to the distinctiveness of the learned-contexts. Although the contexts are very distinct (the inter-context spacing is five simulated steps) there is still a high degree of similarity between contexts in neighbouring serial positions. If the context distinctiveness was increased further, this effect could be reduced, however this would also reduce the degree of bowing in the serial positon curve which occurs naturally as a result of the learned-context similarity.

Also, performance for the confusable condition is better later in the list than is evident in the empirical data. This is attributed to the artificial manner in which item similarity is simulated. In this case, each item is 50% similar to every other (i.e. they share 50% of their features in common).

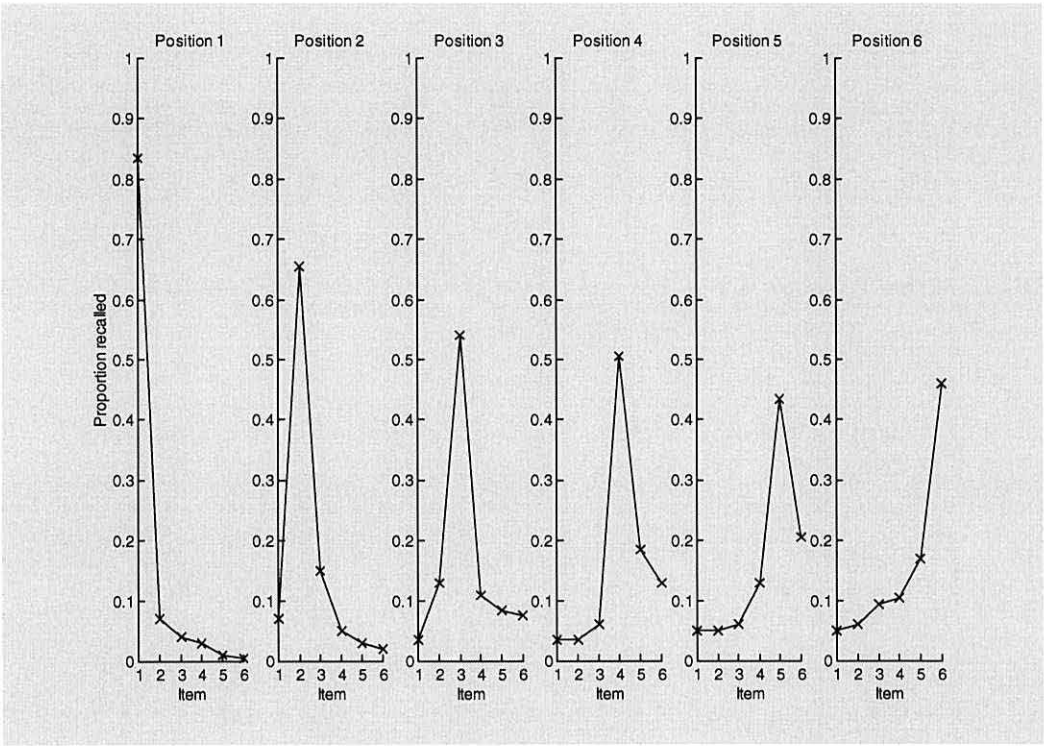

Figure 7.7 *Transposition matrices for six item lists of confusable items (Adapted from Henson, Norris, Page & Baddeley, 1996)*

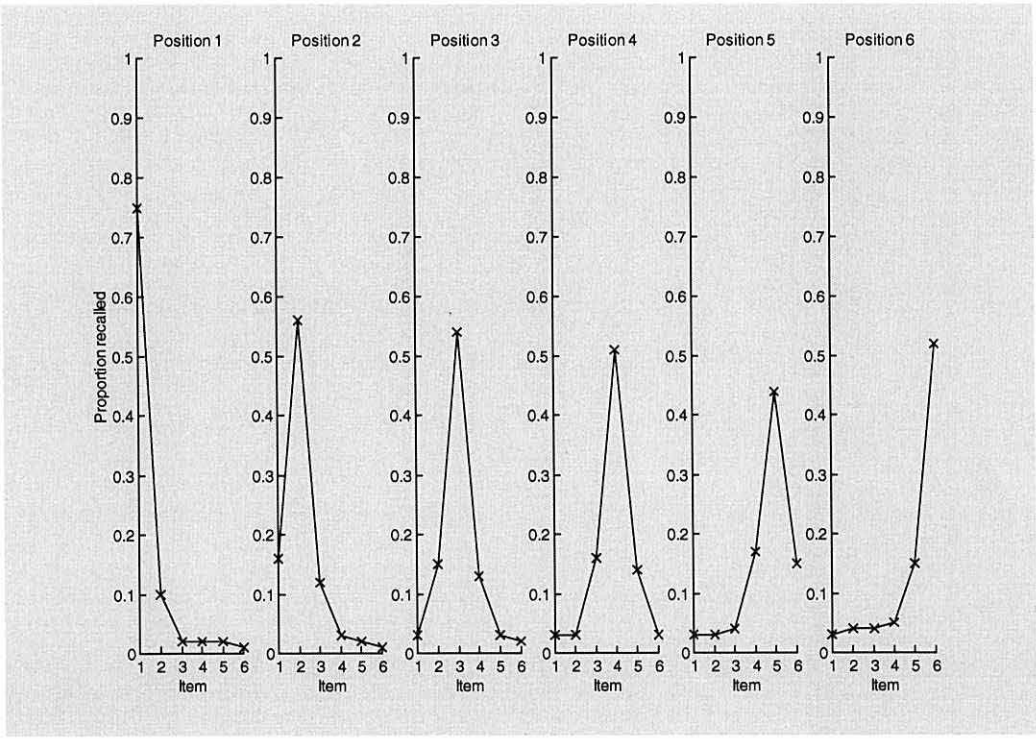

Figure 7.8 *OSCAR transposition matrices for six item lists of confusable items*

In order to increase errors due to item similarity, the degree of similarity would need to be increased. Unfortunately, when the similarity is increased to 75%, performance degrades in each of the target serial positions. It appears therefore, that it would be prudent to develop a vocabulary of items with a more expansive range of similarity (e.g. 66% similarity; cf. TODAM; Baddeley, Papagno & Norris, 1991).

However, these findings confirm that increasing item similarity introduces order errors. Comparison of figures 7.6 and 7.8 reveals that although more errors occur in the confusable condition with items immediately adjacent to the target item, the difference is greater for items further away from the target item. This is most evident when comparing the proportion of items recalled in the sixth position. In the nonconfusable condition, the first three items are each recalled 1% of trials. However, in the confusable condition this increases to approximately 3.5% for each item.

However, the results are further compounded by the inclusion of the lexicon of distractor items. This introduces intrusion errors and will have reduced the number of order errors that might have occurred had no distractor items been available during recall.

7.7 Simulation 28: *Phonemic similarity effect for alternating lists*

Introduction

The phonemic similarity effect for alternating lists (Baddeley, 1968, experiment 5) is an important empirical finding as it contradicts what would be predicted by chaining based accounts of short-term memory (e.g. Wickelgren, 1965a). This is reflected in the number of recent chaining based models of serial order that have failed to reproduce the effect. These include TODAM (Baddeley, Papagno & Norris, 1991) and the original network model of the articulatory loop (Burgess & Hitch, 1992).

Briefly, Baddeley (1968, experiment 5) demonstrated that if a sequence of alternately phonemically confusable and nonconfusable items were presented visually to subjects, errors occurred between the confusable items and not the nonconfusable items. This

finding contradicts what would be expected if short-term memory were chaining based. If this were the case, then a series of six items, for example, a *confusable-nonconfusable* list (C_1, NC_1, C_2, NC_2, C_3, NC_3) could be considered instead as five cue-target item pairs (C_1-NC_1, NC_1-C_2, C_2-NC_2, NC_2-C_3, C_3-NC_3). A chaining based account would predict that if, for example, an error occurred in the first serial position and C_2 was recalled in favour of C_1, this would now become an incorrect cue for the next pair. Therefore there is every chance that NC_2 would be recalled, as the accurate response to that cue, instead of NC_1 which would in fact be the correct response for that serial position. If this were the case, and *cue* errors were occurring, then the nonconfusable items would suffer the most.

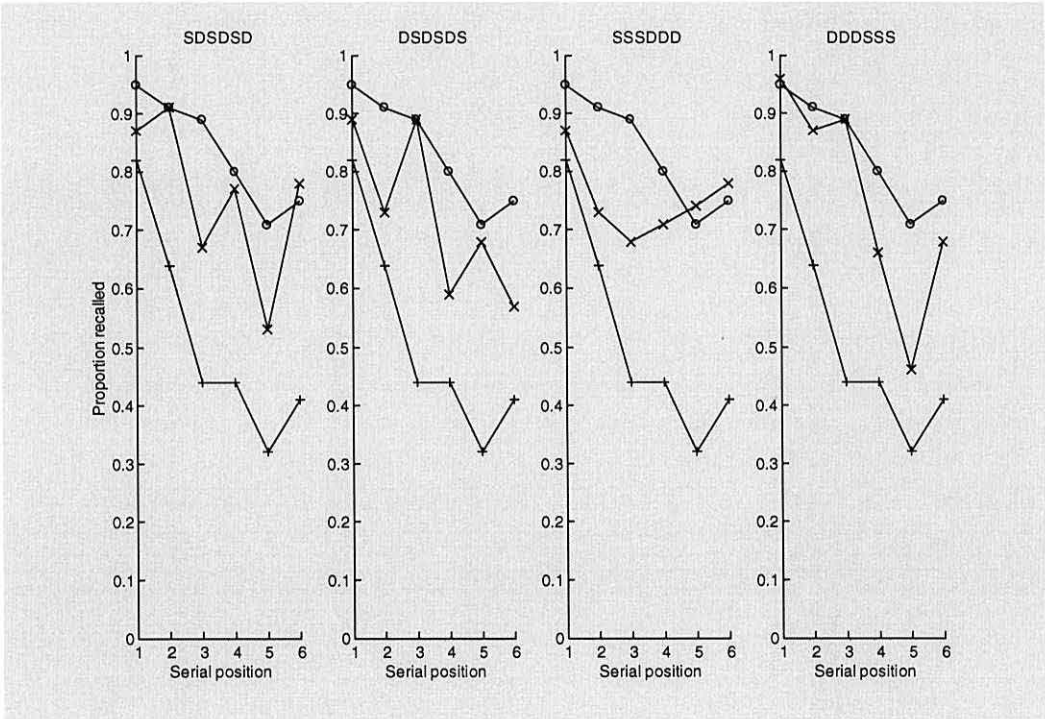

Figure 7.9 *Serial position curves for the alternating list conditions of Baddeley's phonemic similarity experiment (1968, experiment 5)*

In fact, Baddeley's results (figure 7.9) illustrate that the converse is occurring: errors only occur for phonologically confusable items. This leads to the conclusion that order errors must occur during the retrieval process. Using the previous example once more, when C_2 is recalled in favour of C_1, retrieval of NC_1 does not suffer as C_2 is not used as the cue for the second serial position. However, recall of the next confusable item *will* suffer as recall of the same item in a second serial position is prevented i.e. order

errors are occurring during retrieval. Therefore, recall of the nonconfusable items will not be affected by performance for the confusable items as is confirmed by Baddeley's results. Henson, Norris, Page and Baddeley (1996) refer to this property, that order errors only occur between phonemically-similar items, as the *similarity constraint*. In the following simulation, OSCAR is applied to the problem of modelling Baddeley's phonemic similarity effect.

Method

This simulation employed the same procedure and parameters as were used for experiment 23. However, in an attempt to replicate Baddeley's study, OSCAR is presented with a sequence of six items in one of six different arrangements: six nonconfusable items (*DDDDDD*); six confusable items (*SSSSSS*); alternating confusable and nonconfusable items (*SDSDSD*); alternating nonconfusable and confusable items (*DSDSDS*); three confusable followed by three nonconfusable items (*SSSDDD*); three nonconfusable followed by three confusable items (*DDDSSS*). Further, in compliance with Baddeley's simulation, the six items were selected from a vocabulary of twelve (i.e. for the *DSDSDS* list, the lexicon included the learned items and a further six unlearned *DSDSDS* items). Confusable items shared 75% similarity and all model parameters were held constant for the six conditions in this simulation.

Results

Figure 7.10 illustrates OSCAR's attempt to fit the four arrangements of confusable and nonconfusable items described by Baddeley.

The nonconfusable-only serial position curve is identical to that presented in simulation 23, with extended primacy and last item recency due to the edge effects. The confusable-only serial position curve is less well defined than the nonconfusable-only condition. There is extended primacy and a trough which occurs in the fifth serial position, however the last item recency effect is only approximately 2%. Initially it would appear that OSCAR does in fact replicate Baddeley's data. In each of the first two cases (*SDSDSD* and *DSDSDS*), the "sawtooth" curve is bounded by the

nonconfusable-only and confusable-only serial position curves in keeping with the experimental data.

Performance for phonologically confusable items in the alternating list conditions is poorer than for the nonconfusable items in accordance with Baddeley. However, recall of the nonconfusable items is poorer in the alternating list conditions than in the nonconfusable-only condition which contradicts the prediction in section 7.7.1.

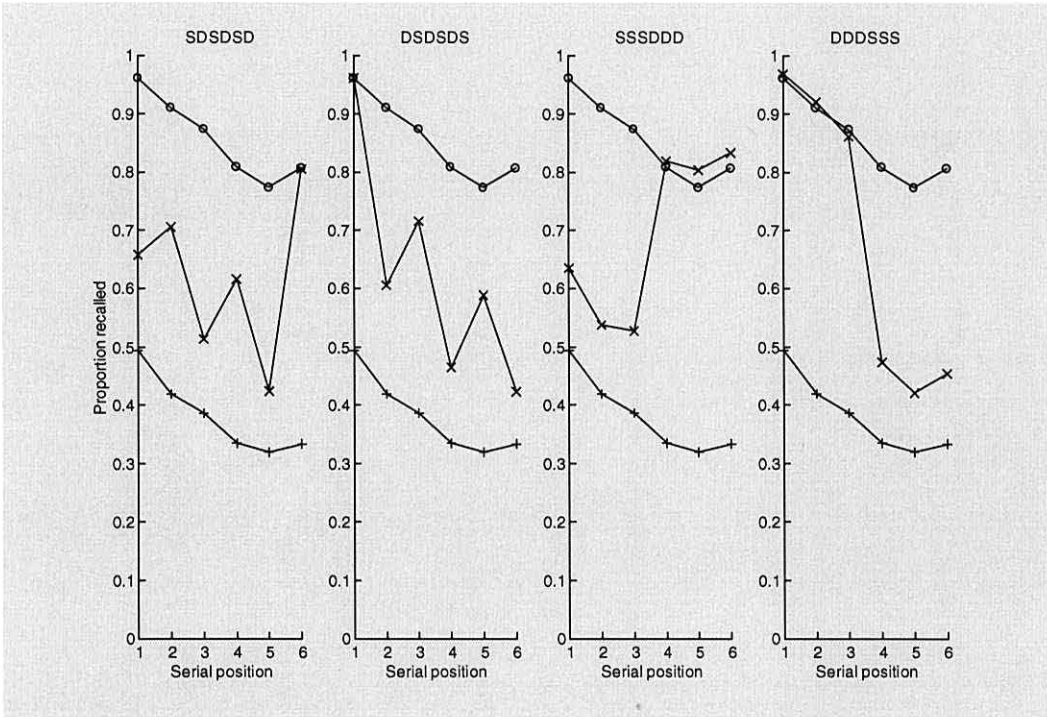

Figure 7.10 *Serial position curves for OSCAR's fit to the alternating list conditions of Baddeley's phonemic similarity experiment (1968, experiment 5)*

In the second pair of results (the *SSSDDD* and *DDDSSS* conditions), the similarity to the empirical data is more apparent. Performance is again bounded by the nonconfusable-only and confusable-only lists, however in this case the nonconfusable items in both conditions are unaffected by the presence of the confusable items in the list. In order to facilitate an evaluation of the performance of OSCAR in this task, the transposition matrix for the nonconfusable, confusable (*DSDSDS*) list from simulation 27 is presented in figure 7.11. Note that performance is higher than that presented here primarily as a result of the set of more distinct contexts used in that simulation and the less confusable stimuli (50% overlap).

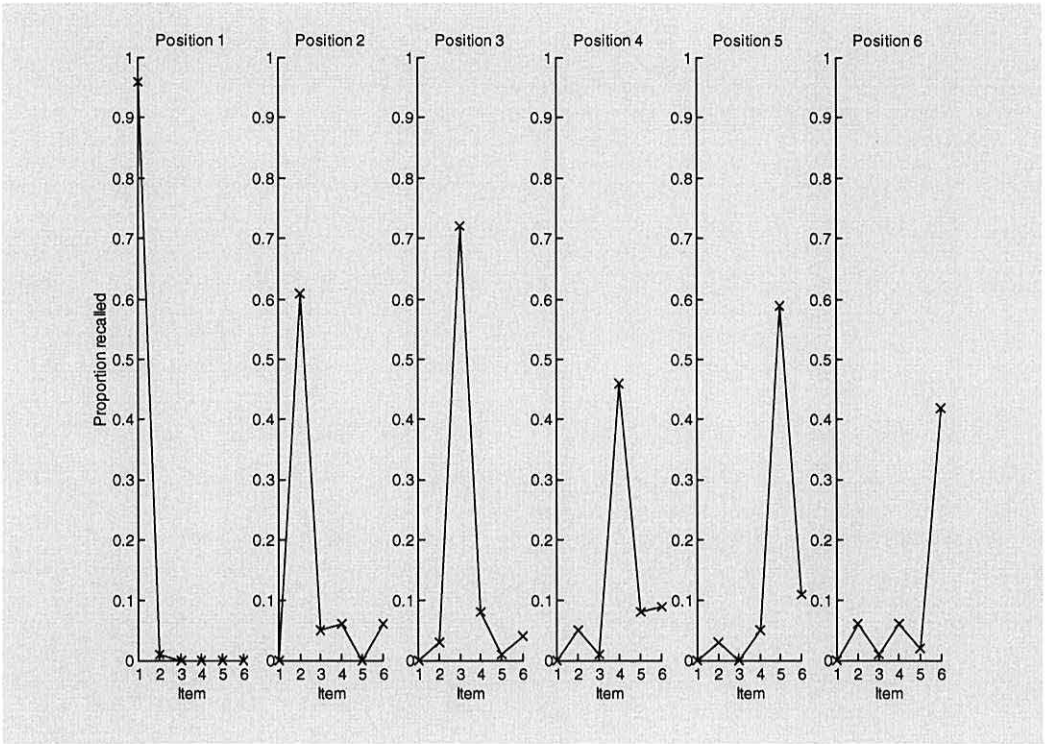

Figure 7.11 *OSCAR transposition matrices for six item list of nonconfusable, confusable items (Data taken from simulation 27)*

Figure 7.11 reveals that recall in the first serial position is unaffected by the presence of the confusable items in the list. The first item is recalled correctly in 96% of trials in the target serial position. The second item is recalled in 1% of trials, while none of the other learned items are recalled at all (n.b. this implies that the remaining 3% of recalls may be classified as intrusion errors).

However, in the second serial position, the first position where the target item is confusable, the target item is recalled approximately 61% of the time. The distribution of errors about the target item in the second serial position warrants close examination. The first item is not recalled in the second serial position while the third item is only recalled in 5% of trials. This is in stark contrast to the large proportion of recalls involving items from the serial positions immediately adjacent to the target item in the confusable-only condition of the previous simulation. However, we would expect, in accordance with the locality constraint observed in the previous simulation, that the fourth, fifth and sixth items would be recalled only a very small proportion of

trials. This is clearly not the case as the fourth and sixth items are both recalled in 6% of trials each, accounting for 41% of all the errors that occur in the second list position (again we note that 12% of errors are distractor item intrusions). A similar effect is observed in the fourth, and most strikingly in the sixth, serial position.

Discussion

Before attempting to draw a conclusion from these findings, it is important to first summarise the key results. OSCAR was applied to the problem of replicating the phonemic similarity effect illustrated by the Baddeley (1968, experiment 5). OSCAR produced serial position curves using identical parameter settings for four different list conditions: alternately confusable and nonconfusable (*SDSDSD*); alternately nonconfusable and confusable (*DSDSDS*); a confusable triple followed by a nonconfusable triple (*SSSDDD*); and finally, a nonconfusable triple followed by a confusable triple (*DDDSSS*). Recall of each learned list was influenced by the inclusion of a lexicon of unlearned items at recall and an inhibitory mechanism preventing repeated items.

The results are presented in the serial position curves of figure 7.10 and the distance functions of figure 7.11. Examination of the serial position curves reveals that performance for each of the four conditions is bounded by the nonconfusable-only and confusable-only conditions. Recall for the list of confusable items is poorer that for the list of nonconfusable items (Conrad, 1964). For the alternating list conditions, this is also the case, and results in the characteristic "sawtooth" shape of the serial position curves with confusable items occupying the troughs and nonconfusable items, the peaks of the curve. However, in contrast to the empirical data, nonconfusable items suffer in the alternating list conditions. Henson, Norris, Page & Baddeley (1996) observe that a similar effect is found in the empirical data as knock-on effects: poorer performance for nonconfusable items that follow an erroneously recalled confusable item. This is not the case for the *SSSDDD* and *DDDSSS* conditions where performance mirrors that of the empirical data.

In these conditions, order errors are clustered within the phonemically-like groups. For example, at the boundary condition where the last confusable item is recalled in the third serial position and the first nonconfusable item is recalled in the fourth position, errors only involve items from the same phonemically-like groups. Within these groups, however, edge effects mean that the first and third items are recalled better than the central item. This is evident as bowing in both the nonconfusable portion of the *SSSDDD* list condition and the confusable portion of the *DDDSSS* list condition. However, in the alternating list conditions, this effect is not as apparent.

Analysis of figure 7.11 (generated during simulation 27) reveals that in these alternating-list conditions, order errors also involve items from the phonemically-like groups. This is illustrated most clearly by the order errors that occur in the sixth serial position where the target is a confusable item. The nonconfusable items are recalled in decreasing amounts as the distance between them and the target item increases (i.e. item five is recalled in 2% of trials, item three in only 1% and item one not at all). However, the confusable items are both recalled in 6% of trials.

Clearly errors occur because of two different mechanisms controlled by OSCAR. The first, which affects cueing, is the distinctiveness of the context vectors. We have demonstrated (simulation 11) that reducing the distinctiveness of the contexts results in poorer serial order performance across each serial position. Figure 7.11 confirms that even in the alternating-list conditions, the proximity constraint still applies and the majority of order errors do involve items that are immediately adjacent to the target item. However, the second mechanism, which affects retrieval, is the similarity of the confusable stimuli. This too has been shown to reduce performance when the confusability is increased (simulation 21).

Further, this analysis is supported by consideration of the transposition matrix of figure 7.11. This confirms that errors occur due to the similarity of neighbouring contexts: i.e. target items are recalled in decreasing amounts in the serial positions adjacent to the target position. This is very apparent for the nonconfusable items in the odd serial positions. However, in the even serial positions occupied by the confusable serial

positions, further errors are caused (across a greater spread of incorrect positions than in the nonconfusable condition) by errors during retrieval where the retrieved item is incorrectly recalled as one of the incorrect confusable items. This is apparent in the subtle increase in the proportion of errors in the serial positions where the target is a confusable item than in the position nearby where the target is distinct.

In conclusion, superficially, OSCAR appears to have replicated the phonemic similarity effect reported by Baddeley (1968, experiment 5) in that the serial position curves presented in figure 7.10 do replicate the general form of Baddeley's (Figure 7.9). This confirms the hypothesis that errors due to (phonemic) similarity are occur during the retrieval stage. However, closer examination of the transposition matrix reveals that the errors occurring at retrieval are in addition to errors occurring at cueing due to the proximity and hence similarity of the context vectors. These errors are localised to the positions immediately neighbouring the target serial position and result in a degradation of performance most noticeably for the nonconfusable items.

In fact, simulations in which the spacing between the list items has been varied have illustrated that it is possible for OSCAR to produce the reverse effect to that reported here: i.e. supporting the prediction by the chaining account that nonconfusable items suffer the most in recall tasks of this nature.

7.8 Simulation 29: *Item and order errors*

Introduction

In a series of experiments, Healy (1974) attempts to separate item and order errors. In the first simulation, the *Order Only Experiment*, Healy asked subjects to remember only the order in which four items were presented. This was made possible by repeatedly presenting the same four items in one of 24 different orders in order to eliminate any *item* errors. In the second experiment[14], the *Item Only Experiment*, subjects were requested to recall only the items that were presented. This was implemented by presenting subjects with a set of three items for each serial position

[14] The *all-different context* condition.

and therefore providing the subjects with all order information during recall. By eliminating *order* errors in this manner, Healy could examine item effects in isolation. Her results, presented in figure 7.12 confirm that bowed serial position curves are produced in the *order* only experiment and not for the *item* only experiment.

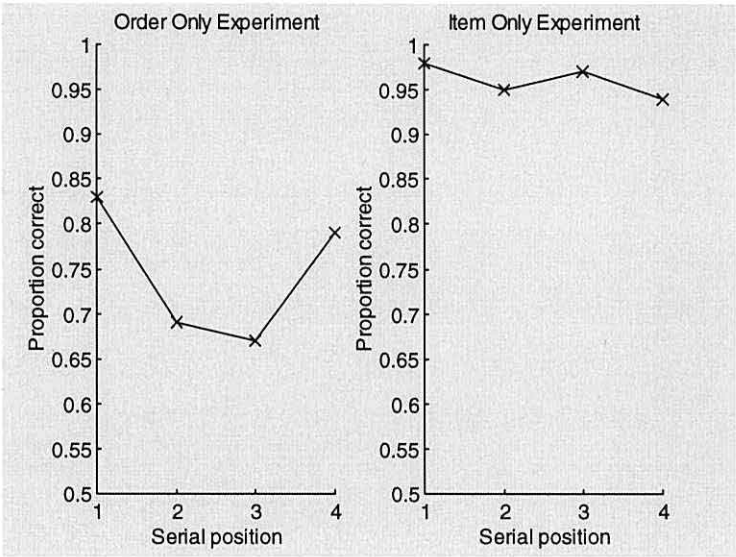

Figure 7.12 *Order only and item only data for four item list*
(Adapted from Healy, 1974, figure 1 and figure 3, three digit delay condition)

Furthermore, examination of the distance function for the order only simulation confirms that the probability of a letter being replaced by another decreases as the distance between them increases (figure 7.13). In the following simulation, we aim to replicate these findings and separate item and order information for recall of a four item list.

Method

This simulation employed the same procedure and parameters as were used in simulation 23 but for the following variations. The context distinctiveness was controlled by reducing the inter-context spacing to four. No noise was introduced to the network and recall was inhibition free. Also, in order to replicate Healy's *Order Only Experiment*, OSCAR was presented with a list of four items (selected from a vocabulary of 12). However, a more complex approach was required to replicate the *Item Only Experiment*: Healy reports that each subject is given a choice of three items

at each serial position. Therefore, in the following item only simulation, OSCAR was presented with three items (the target and two distractors) for each serial position during recall. The distractor items are different for each serial position. The free parameters were held constant for both the *item only* and *order only* conditions.

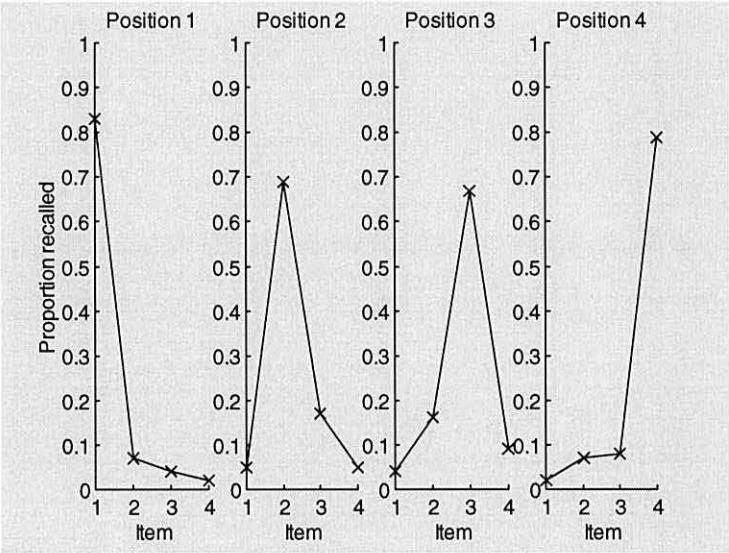

Figure 7.13 *Distance functions for the order only simulation (Adapted from Healy, 1974, figure 2, three digit delay condition)*

Results

The serial position curves for both the *order only* and *item only* simulations are presented in figure 7.14. These confirm that only the *order only* serial position curve exhibits the bowing (performance dropping from 78% for the first serial position to 58% and 56% for the central positions, before rising again to 72% for the fourth item). The *item only* serial position curve is almost level for each serial position: 98% for the first item and approximately 96% for each of the other positions. Both the *order only* and *item only* serial position curves compare favourably with the empirical data (cf. figure 7.12).

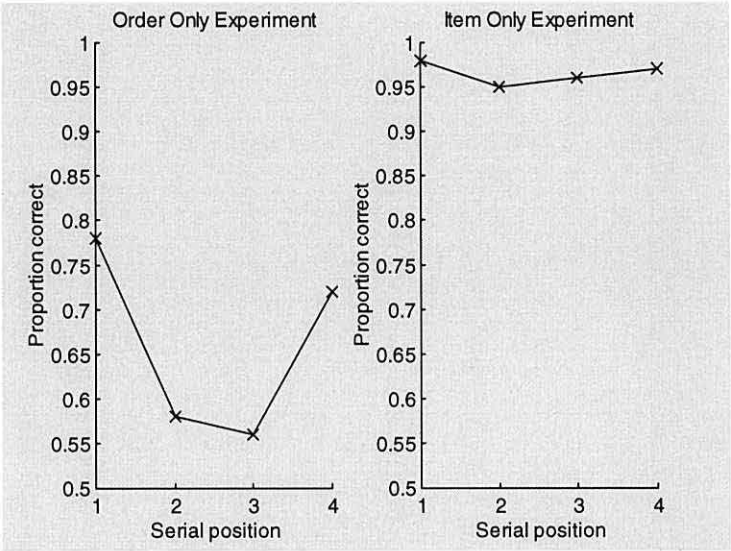

Figure 7.14 *OSCAR serial position curve for order only and item only simulations*

More complex analysis is possible if the distance function for the order only simulation is analysed separately. The distribution of order errors is clearly visible in figure 7.15 and, once more, OSCAR's simulation compares favourably with the empirical data (figure 7.13). It is clear that items are recalled most often in their target serial position and the probability of them being recalled outside this position decreases sharply as the displacement from the target serial position increases (in either direction).

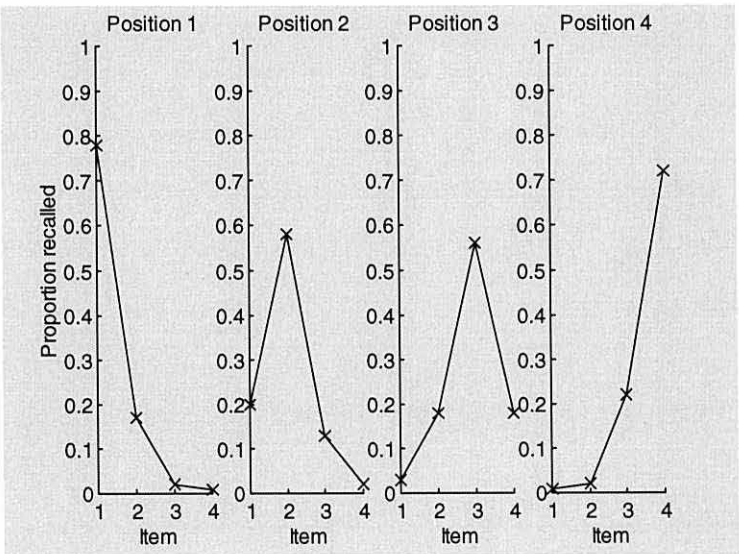

Figure 7.15 *OSCAR distance functions for the order only simulation*

Discussion

Comparison of the both the general form of the item and order serial position curves (figure 7.14), and the error distribution of order errors (figure 7.15), with the empirical data presented by Healy (figures 7.12 and 7.13) reveals that OSCAR provides an adequate fit to the data. Although OSCAR is performing with a slightly lower degree of accuracy during the order only simulation, it still reproduces the symmetric error distribution in the distance function.

Healy concludes from these findings that as the serial position curves for both the *order only* and *item only* conditions are different, so different mechanisms, isolated by the different conditions employed in this simulation, must be responsible for the error distributions. Estes (1972) suggests that the *order only* serial position curve is bowed because the probability of a transposition between one item and another decreases as the items become more widely separated. Therefore, as the second and third items have more neighbours, so they have a higher probability of being recalled out of order, which is reflected in the poorer performance for the central positions. *Item only* errors are accounted for by considering that these errors will only occur due to degradation of the item elements (features) and as the probability of this occurring can be assumed to be constant for each item, so the *item only* serial position curve should be flat (Estes, 1972).

This analysis is confirmed by OSCAR's performance. In the *order only* simulation, errors are occurring at *retrieval* due to the similarity of the cue context with those in the neighbouring serial positions: there is more chance of the context confusing with those in the serial position immediately adjacent to the target serial position which results in the bowing of the serial position curve. Therefore, *order only* performance could be reduced by either reducing context vector distinctiveness or by using a vocabulary of more confusable items.

In the *item only* simulation, retrieval errors are eliminated as only the target item and two distractor items are available at redintegration for each serial position. We demonstrated in simulation 26 how increasing the number of unlearned distractor

items available at recall has little effect on performance (cf. Drewnowski, 1980). Therefore, all of the errors occurring in this simulation are due to the inaccuracies of the associative and deblurring mechanisms. The context distinctiveness has no influence on the ability to recall *item only* information. Furthermore, it is predicted that if a vocabulary of similar items were used (i.e. each of the four learned items *and* each of the twelve distractor items were 50% similar) so retrieval errors could be introduced into the system, resulting in a curve that would still remain flat, but with a lower level of performance overall.

Clearly OSCAR provides a single mechanism that is capable of reproducing the different item and order behaviour illustrated by the empirical data.

7.9 Simulation 30: *Partial reinstatement of context*

Introduction

This simulation aims to illustrate the effects of recalling serial position information from a perspective near to the end of the list. All the simulations described thus far require ordered recall of the whole list of items. As a result, there is no overlap between the retrieval context and the learned-context for items late in the list. However, if these late items could be recalled first they would be at an advantage as the learned-context would not have evolved far from the state it was in when those items were learned. This, we predict, would benefit the late items and introduce recency into the serial position curve.

In order to compare OSCAR with an empirical result, a probed recall task for visually presented stimuli is presented (Murdock, 1968, experiment 3). During probed recall, subjects are requested to recall the item that occurred in the nth serial position immediately after recall. A different list is then presented and recall for a different serial position requested. Murdock presented subjects with lists of five, eight or eleven items and demonstrated that in each condition there was a strong recency effect (figure 7.16). The extent of this recency effect is in sharp contrast to that exhibited by complete serial recall where the last item recency effect is typically very small (cf.

figure 7.1). These results are consistent with the idea that the context at recall is more similar to that for the most recently learned items, which is reflected in improved recall for the last items resulting in an increase in recency.

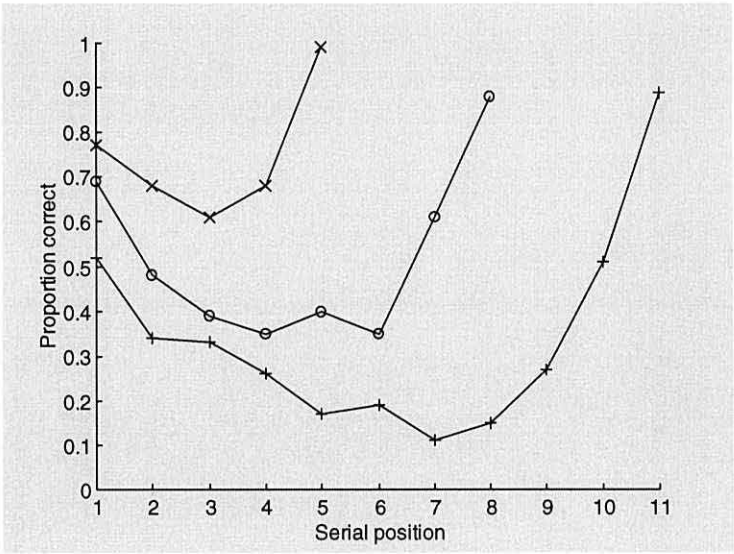

Figure 7.16 *Proportion of recall as a function of serial position (ordinal probe) (Adapted from Murdock, 1968, figure 3)*

In the following simulation, OSCAR attempts to replicate the probe recall serial position curve of Murdock (1968, experiment 3).

Method

This simulation employed the same procedure and parameters as were used for simulation 23 but with the following variations. The context distinctiveness was controlled by reducing the inter-context spacing to four. Forgetting was provided by the weights decay parameter (0.80) and output interference (10% noise). Performance was reduced overall by introducing a small amount of noise during learning (10% noise). No vocabulary of distractor items was added to the lexicon of recallable items and no inhibitory mechanisms were used during recall. The number of list items was varied in order to replicate the empirical data.

However, without additional controls, the model would perform recall by reinstating the learned-contexts completely. However, we have stated that in order to model these effects, partial reinstatement of contexts is required. We model this with OSCAR by

ensuring that a small proportion (20%) of the reinstated context probes are unable to be reinstated and are instead identical to the context at the end of the list.

Results

The results of this simulation are presented in figure 7.17. Clearly partial reinstatement of contexts introduces a strong degree of recency into the serial position curve. This is in stark contrast to the serial recall of visually presented items of the previous simulations.

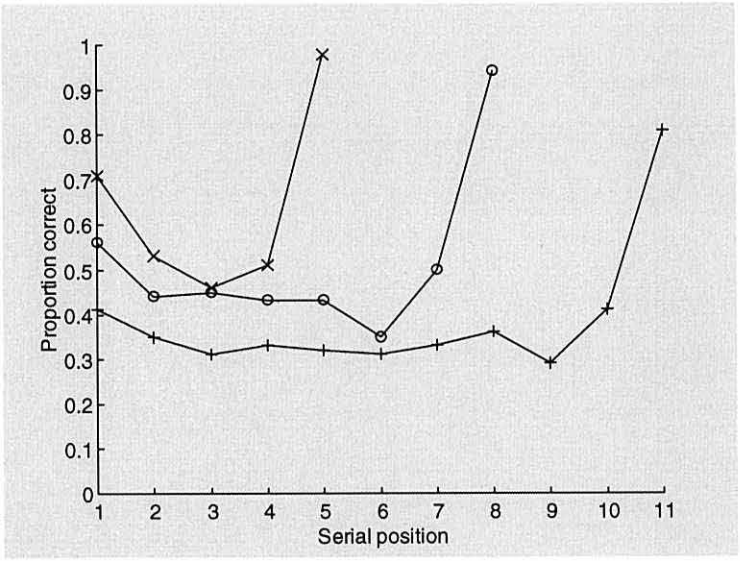

Figure 7.17 *Serial position curves for five, eight and eleven item lists with only partial reinstatement of learned contexts*

Discussion

Although the fit provided by OSCAR to Murdock's (1968, experiment 3) data is not accurate, as it lacks the bowing and extended primacy gradient of the empirical data, the curves do possess the large proportion of last item recency. Clearly partially reinstating the learned-contexts by ensuring that a proportion of it is identical to the state of the context at the end of the learning stage, improves recall of the last couple of items. This result is in sharp contrast to the serial order recall data presented previously where complete reinstatement of the learned contexts was possible and recency effects were slight.

7.10 Conclusion

In this chapter, we have attempted to replicate a range of empirical data using OSCAR. Here, we briefly summarise each simulation and present a summary of the parameter values used in each.

Initially, OSCAR was fitted to Baddeley's (1968, experiment 5) serial position curve for serial recall of visually presented stimuli. OSCAR demonstrated the extended primacy and last item recency apparent in the empirical data. In the second simulation, OSCAR attempted to replicate the memory span functions for digits, letters and words of Crannell and Parrish (1957). This it was able to do by manipulating the strength with which each item was learned. OSCAR was found to have a memory span for digits of approximately 7.1 items, in line with the empirical data.

Modelling phonemic similarity by allowing a proportion of vector elements to be shared between items, it was possible to investigate the effect of phonemic similarity in a number of conditions. Simulation 25 illustrated that memory span for confusable items was poorer than for nonconfusable items (Baddeley, 1966; Conrad, 1965; Wickelgren, 1965a).

In the next simulation, the effect of increasing the number of recallable items available during retrieval, by adding unlearned distractor items to the vocabulary, was reported and compared with that found by Drewnowski (1980). In accordance with the empirical data, it was found that increasing the number of recallable items had little effect on the proportion of items recalled correctly for a number of different list lengths.

In the first of three simulations addressing item and order errors, simulation 27 compared the distance function, or transposition gradients, generated by OSCAR when recalling lists of six nonconfusable or confusable items, with those presented by Henson, Norris, Page and Baddeley (1996). Distance functions highlight the distribution of order errors across each serial position illustrating how order errors

mainly involve items occupying immediately adjacent serial position. OSCAR reproduced the form of Henson et al.'s curves with the extended primacy, small last item recency and localised distribution of order errors (Estes, 1972).

Next, OSCAR was applied to the problem of matching Baddeley's (1968, experiment 5) phonemic similarity data for alternating confusable and nonconfusable lists. This result illustrates that when lists of alternately confusable then nonconfusable (or vice versa) stimuli are learned, confusable items suffer most during recall. This contradicts the prediction made by chaining-based hypotheses of memory that predict that nonconfusable items suffer due to cueing errors. This is reflected in the inability of chaining based models such as TODAM (Lewandowsky & Murdock, 1989; Baddeley, Papagno & Norris, 1991) and the original network model of the articulatory loop (Burgess & Hitch, 1992) to reproduce the result. OSCAR replicates the general form of the curves, although in contrast to the empirical data, performance for the nonconfusable items is reduced in the alternating list conditions.

Simulation 29 is an attempt to replicate Healy's (1974) experiment that examined the distribution of item and order errors. OSCAR is adapted in order to facilitate the modelling of this data and replicates the finding that order errors result in a bowed serial position curve while item errors occur equally across each serial positions. This result is significant as it demonstrates that OSCAR's learning and recall mechanism is capable of reproducing the different distributions of item and order errors.

Finally, the effect of partially reinstating the learned-context on OSCAR's ability to perform serial recall is considered. It is proposed that by overlapping the reinstated contexts with the state of the learned-context when the last item has been learned, recall will favour the last few items. OSCAR is fitted to data from a probed recall for visually presented stimuli task (Murdock, 1968, experiment 3). In accordance with the empirical data, OSCAR's serial position curves exhibit a huge recency effect, however the primacy component of the curve is poorer than in Murdock's experiment.

Table 7.2

Summary of parameter values for each simulation of the empirical data presented in this chapter

	Simulation							
Parameters	23	24	25	26	27	28	29	30
Number of contexts	20	20	20	20	20	20	20	20
Context dimensionality	16	16	16	16	16	16	16	16
Context distinctiveness	5	4	4	5	5	5	4	4
Number of items	6	var	5	var	6	6	4	var
Similarity of items (%)	n/a	n/a	75	n/a	50	75	n/a	n/a
Number of distractor items	6	0	0	var	6	6	var	0
Weights decay	0.9	0.87	0.87	0.9	0.9	0.9	0.9	0.8
Non-linear learning rate scale	0.9	var	0.9	0.9	0.9	0.9	0.9	0.9
Learning noise (%)	20	10	10	20	20	20	0	10
Recall noise (%)	20	10	10	20	20	20	0	10
Output threshold	0	0	0	0	0	0	0	0
Inhibition preventing repeats	yes	no	no	no	no	yes	no	no

In conclusion, OSCAR has demonstrated the ability to fit a number of fundamental serial order paradigms with the minimum of free parameters. These parameters, and the values they are assigned, are summarised in table 7.2. Although in some cases, the fit to the empirical data is not as close as might be desired, OSCAR has been shown to possess the majority of the characteristics of each empirical paradigm.

CHAPTER 8

General Discussion

8.1 Introduction

The following chapter summarises the findings of this thesis. In the first section, we consider the respective advantages and disadvantages of connectionist and mathematical models of memory. Next, we introduce the developmental connectionist architecture, DARNET. In section 8.4 we present the OSCAR model of serial order. We review its architecture and compare it with other formal models of serial order. Section 8.5 examines the success of OSCAR, in comparison with other formal models of memory, in addressing a range of empirical data. Finally, we suggest that further research should integrate the DARNET and OSCAR architectures to produce a developmental account of serial order.

8.2 Comparison of model architectures

Recent models of immediate memory have fallen into two categories: the first are mathematically-based models such as CHARM (Metcalfe Eich, 1982) and TODAM (Murdock, 1982); the second are connectionist models, including the competitive queuing model (Houghton, 1990, 1994a), recurrent networks (Jordan, 1986) and the network model of the articulatory loop (Burgess & Hitch, 1992, 1996). Before developing a novel architecture, these approaches must be evaluated.

Mathematical models

Mathematical models offer the ability to perform single-trial learning. This is an inherent property of their associative mechanism, often the mathematical process of convolution (e.g. Metcalfe Eich, 1982). However, because the ability to perform single-trial learning and recall is hard-wired into the architecture of models such as

CHARM and TODAM, they can not offer a developmental account of learning, nor is clear how they could be extended to do so. Also, association by convolution requires that the dimensionality of the association is proportional to the dimensionality of the stimulus. Because of this, the capacity of a composite memory vector is limited by the dimensionality of the stimuli.

Connectionist models

Connectionist models offer a more developmental approach to learning by implementing gradual error correction learning rules, such as back-propagation (Rumelhart, Hinton & Williams, 1986). Furthermore, connectionist architectures possess the ability to generalise learned behaviour to patterns of previously unseen stimuli (Rumelhart & McClelland, 1986). Also, in contrast to convolution based mathematical models, there is no limit on the capacity of the memory trace (e.g. the number of hidden units in a multi-layer back-propagation network). However, traditional developmental connectionist networks have been unable to perform single trial learning. Where recent models, such as the competitive queuing model (Houghton, 1990, 1994a) and network model of the articulatory loop (Burgess & Hitch, 1992, 1996), have demonstrated the ability to do so, these models have then been unable to provide a developmental account of learning. Connectionist networks are also susceptible to catastrophic interference (McCloskey & Cohen, 1989; Ratcliff, 1990; Lewandowsky, 1991).

Conclusion

The primary objective for developing the DARNET architecture was to provide a developmental account of learning. In their present form, mathematical approaches have to be rejected on the grounds that the ability to perform single-trial learning is already a feature of these models and they are therefore unable to provide a developmental account. Therefore, we developed a connectionist-based network, that employed a gradient descent learning algorithm in order to learn to perform single trial learning.

All that was required of the OSCAR architecture was the ability to perform single trial learning. Therefore, the choice of mathematical versus connectionist-based network was arbitrary, however we opted for a connectionist-based architecture because their capacity is not limited in the same manner as the convolution-based models.

8.3 DARNET

Need for a developmental account

There is much empirical data that examines the development of memory in children (e.g. Gathercole & Adams, 1993; Hitch, Halliday & Littler, 1993; Hulme, Thomson, Muir & Lawrence, 1984; Hulme & Tordoff, 1989). Much of this work is concerned with the development of memory span and rate of articulation. For example, memory span increases with age. Furthermore, the phonemic similarity effect (Conrad, 1964) is more apparent in older subjects (Hulme & Tordoff, 1989). As the current associative models that employ convolution and correlation as their associative mechanism are unable to provide a developmental account of learning, there is need for such an architecture.

DARNET architecture

DARNET, a developmental associative recall network (Brown, Hyland & Hulme, 1994; Brown, Dalloz & Hulme, 1995; Brown, Preece & Hulme, 1995; Brown, Hulme & Dalloz, 1996) is a connectionist network that employs a gradient descent learning algorithm to learn how to associate novel pairs of input vectors in a distributed memory trace. During phase one training, DARNET is presented with novel pairs of normalised vectors of features. The association between the two items is stored in a memory trace of arbitrary dimensionality. DARNET is then presented with one of the items, selected at random, as a probe for recall. The error between the output and the target item (the second of the pair of stimuli) is calculated and used as the basis for the error correction learning algorithm. This alters the learned storage and retrieval weights in order to minimise the magnitude of the error between the retrieved item and the target item. The phase one learning process is repeated until the model has learned to accurately store and recall novel stimuli.

During the phase two learning stage, DARNET stores pairs of novel stimuli in a composite memory trace in order to attempt paired associate learning (e.g. Metcalfe Eich, 1982). Neither the storage or retrieval weights are updated during this phase. DARNET offers a significant advantage over other mathematical models of single trial association such as CHARM (Metcalfe Eich, 1982) and TODAM (Murdock, 1982) as, unlike convolution-correlational based models of association, the capacity of the composite memory trace is not rigorously defined by the learning algorithm. In section 4.3.4, we demonstrated how increasing the capacity of the memory trace during phase one learning can significantly reduce the amount of time required to train the network to an optimal level of performance. During phase two, an increase in memory trace capacity resulted in an improvement for paired associate storage and recall, for both confusable and nonconfusable stimuli (section 4.4.1).

Paired associate learning

It is possible, by storing the storage and retrieval weights of a partially trained DARNET midway through phase one training, to examine the development of paired associate learning (Metcalfe Eich, 1982). In section 4.4.2, we demonstrated that the network can be trained to perform paired associate learning and recall at a level comparable with CHARM. At this level of performance, DARNET demonstrated a similarity effect when presented with confusable stimuli (Conrad, 1964) and produced very few cue intrusion errors. In a second condition, where phase one training was stopped before DARNET was performing optimally, DARNET exhibited many more cue intrusion errors in the similar item condition (section 4.4.3). This provides evidence that a developmental account of associative memory is desirable as the proportion of errors exhibited by the model may reflect its associative ability. It may also be the case that similar developmental effects are exhibited by human subjects (Brown, Preece & Hulme, 1995).

8.4 OSCAR

DARNET provided us with a developmental account of association. However, it was unable to perform serial order learning and recall. In an attempt to address this, a novel

architecture for serial order was developed, OSCAR. Here we describe the model and review its ability to perform serial recall. We compare the model to recent formal models of serial order.

Basic architecture

The oscillator based associative recall model, OSCAR, attempts to overcome the shortcomings of previous models of serial ordered immediate recall. OSCAR aims to provide a neurobiologically plausible distributed single-trial learning model of immediate memory for serial order. The model implements item-to-context association by Hebbian single trial learning. In order to improve the retrieval and deblurring process, items and contexts are represented in the model by normalised vectors of features (Metcalfe Eich, 1982; Goebel & Lewandowsky, 1991).

Dynamic context signal

Examining control signals employed by recent models of memory, such as the two dimensional control of Houghton (1990) and the context of Burgess and Hitch (1992, 1996), it is clear that a control signal must possess a number of properties. The first property is that contexts which represent states that are close together in time must be more similar to each other than those which represent widely separated points in time. This leads to the second property, which is that the context must not repeat itself. In fact, a representation that would prove suitable is outlined by Church and Broadbent (1990). Church and Broadbent describe a vector for interval timing, built up from a series of oscillators, each with a frequency of twice that of the previous oscillator, based around a circadian period.

In section 5.4, a context vector was developed which not only satisfied these requirements but was also more plausible than the context of Burgess and Hitch (1992) as it employed vectors of continuous features and was more powerful than Houghton's (1990) control signal due to its increased dimensionality. The context vector is reinstatable so that during recall, only the first state of the context is required in order to "rebuild" each context vector in the sequence.

Single trial learning and recall

Single trial learning is facilitated by employing Hebbian learning, although alternative single trial learning algorithms would also be suitable (e.g. convolution and correlation). In section 5.5 we demonstrated how items were presented in sequence to the network and were associated with the context vector that corresponded to the moment in time that the item was presented to the network. After a list had been presented, the Hebbian memory matrix was shown to contain a number of associations between list items and the appropriate context vector. Recall involved reinstating the learned-context for each item and presenting it to the memory trace as a recall probe. The retrieved item was compared with a lexicon of available responses and the item that bore the most similarity to the retrieved item, recalled.

Parameter free performance

The basic model was shown to produce bowed serial position curves with primacy and recency components (section 5.6). These occurred as a direct result of edge effects due to the similarity of the context vectors. Furthermore, even in its most basic form, OSCAR demonstrated the ability to produce order errors, distributed in accordance with Estes (1972).

Model parameters

With the addition of a number of free parameters, OSCAR was fitted to a range of empirical data (see section 8.5). Forgetting in the model occurred as a result of weights decay, interference between item-to-context associations in the Hebbian matrix, the distinctiveness of learned-contexts or the failure to reinstate learned-contexts during recall. The weights decay and noise during learning parameters helped to introduce recency into the model. The learning rate parameter decayed non-linearly during learning to reflect the intuition that successive list items in a sequence became progressively less surprising during learning. Output interference and the learning rate parameter introduced primacy to the model. The effects of learning rate and decay were explored in section 6.3.

An option for output inhibition was also available (cf. Burgess & Hitch, 1992, 1996; Houghton, 1990; Page & Norris, 1995) which either prevented the same item being recalled in successive serial positions, or prevented items being repeated in any two list positions. In contrast to some recent models (e.g. Page & Norris, 1995), the effect of inhibition did not decay with list position, although the inhibitory process that prevented the same item being recalled adjacent serial positions could be thought of as a weak implementation of such a process. A noise threshold, above which items were recalled and below which they were omitted, was shown to introduce item omission errors to the network (section 6.7). Clearly these inhibitory and threshold mechanisms are primitive and we suggest that a new implementation of the architecture should consider the use of a competitive filter such as those seen in a number of recent models (Houghton, 1990, 1994a; Burgess & Hitch, 1992, 1996) or an anti-learning process (Lewandowsky & Li, 1994).

There was also control over the number of recallable items available to the model during the retrieval stage. Typically this included only those items that were presented during learning. However, this could be manipulated to include unlearned items which introduced further item errors during recall. In section 6.4.4, we demonstrated that the combined effects of the inhibitory processes and the number of items in the lexicon of recallable items could affect the recency component of the serial position curve.

Comparison with other architectures

OSCAR has little in common with chaining models such as TODAM (Lewandowsky & Murdock, 1989) and recurrent networks (Jordan, 1986; Elman, 1990). These models implement some form of chaining between at least one of the previous items and the next. OSCAR, in contrast, employs only item-to-context association (cf. Burgess & Hitch, 1992, 1996). Furthermore, OSCAR relies upon single trial learning, provided by Hebbian association, whereas recurrent networks require back-propagation in order to learn one sequence of stimuli. It has also been observed that the magnitude of lexicon items available during recall has no effect on OSCAR (Drewnowski, 1980) whereas it is a fundamental component of the TODAM recall

procedure and is responsible for the degree of recency exhibited by the model (Mewhort, Popham & James, 1994; Nairne & Neath, 1994).

OSCAR, like the Houghton (1990) model, employs the use of a fixed-dimensionality learning-context signal that is reinstatable and does not repeat over long periods of time. However, in contrast to Houghton's model, OSCAR's learning-context signal is of much greater dimensionality and hence possesses greater discriminatory power. It is also motivated by research from other domains (Church & Broadbent, 1991; Gallistel, 1990).

Page and Norris (1995) observe that the primacy gradient, when it is first reinstated before any noise or decay occurs, is akin to the activation of each item in a item-to-context model, such as OSCAR, when the first context is presented to the network. However, the author would argue that in OSCAR's case, unlike the primacy model which would appear to require that every list item possesses a positive activation (although this does suggest apriori knowledge of the list length), the degree of activation when OSCAR is cued will depend on the distinctiveness of each of the contexts in the sequence. If the contexts are "less-distinct", we would expect each item to become active, the first item most becoming the active and then less so for each of the remaining items. However, if each context is "highly-distinct" then we would anticipate that the presentation of the first context would only activate the first few items and not every list item.

Page and Norris (1995) also describe the addition of a second stage of processing which facilitates the modelling of phonemic similarity effects. In particular, they describe the addition of a second primacy gradient to ensure that confusions within the phonemically-like groups obey the transposition locality gradient observed empirically (Henson, Norris, Page & Baddeley, 1996). A similar effect occurs naturally with the OSCAR architecture. For example, if one considers the recall of the first confusable item in the sequence NC_1 C_1 NC_2 C_2 NC_3 C_3, then clearly the context cue for C_1 will be more similar to that for C_2 than for C_3. Clearly, this eliminates the need for any mechanism such as that employed by Page and Norris.

Limitations of the OSCAR model

The OSCAR model, as defined in the current thesis, fails to provide an account for a range of data.

It is unclear how the model could account for rehearsal effects (cf. Baddeley, 1986), although the author would suggest that rehearsal of stimuli items and their respective contexts may have the effect of making the learned contexts more distinct, and hence recall may require only partial reinstatement of the learned contexts for an adequate level of performance. A second suggestion might be that items are rehearsed at a slower rate than that during presentation, and hence each item may become associated with a more highly distinct context during this phase.

The model also fails to demonstrate any long term learning effects (e.g. Hebb effect (Hebb, 1961; cited in Lewandowsky & Murdock, 1989)). As effects such as these have been modelled by previous models of serial order, (e.g. TODAM: Lewandowsky & Murdock, 1989, p40), it is important that the model be expanded to include a long term component in order to do so.

Furthermore, it is unclear how the OSCAR model could account for the effect that it is easier to recall items forwards rather than backwards.

8.5 Accounting for the empirical data

There is a wide range of empirical data that illustrates the behaviour of immediate memory. During a typical serial order task, subjects are presented with a sequence of stimuli such as letters (e.g. Baddeley, 1968) or digits (e.g. Conrad, 1959). These may be presented either visually (e.g. Conrad, 1965) or verbally (e.g. Jahnke, 1963). Recall is typically immediately after learning. Many of the models introduced in chapter 3 have been fitted to data such as that presented here. Their ability, or otherwise, to account for these results exposes the strengths and weaknesses of each model. Here we compare the ability of these models with OSCAR to account for a range of empirical data.

Serial position curve

The serial position curve has been replicated by the majority of the models of serial order. Lewandowsky and Murdock (1989) fit the chaining-based TODAM to a number of serial position curves successfully (Lewandowsky & Murdock, 1989, figure 15). However, Nairne and Neath (1994) argue that the recency component of the TODAM serial position curve is a direct result of the sampling without replacement scheme employed by TODAM during recall. Burgess and Hitch (1992) also attempt to fit the network model of the articulatory loop to serial position curves. However, although the curves have a primacy component both when the model is fully chaining or fully context driven, neither produces a recency component without adaptation of the model (Burgess & Hitch, 1992, p. 454). Page and Norris (1995) also provide a reasonable series of serial position curves, firstly with the most basic primacy model as a direct result of end-effects (Page & Norris, 1995, figure 2), but later providing a good fit to both Baddeley (1968) and Henson, Norris, Page and Baddeley (1996) with the four parameter primacy model (Page & Norris, 1995, figure 6).

Although the ability to produce a symmetric serial position curve occured naturally with OSCAR as a consequence of the end effects (figure 5.15), OSCAR also demonstrated the ability to reproduce a serial position curve possessing both the extended primacy and last item recency of immediate recall of visually presented stimuli (Baddeley, 1968). Only the addition of learning rate and both decay and noise during learning and recall was required (figure 7.1).

Memory span function

Lewandowsky and Murdock fit the open loop version of TODAM successfully to Crannell and Parrish's (1957) memory span functions (Lewandowsky & Murdock, 1989, figure 12). Burgess and Hitch (1992) provide a range of adequate memory span data (Burgess & Hitch, 1990, figures 5-8) and also a function of articulation rate against memory span (Burgess & Hitch, 1990, figure 7) with a model using a combination of the chaining and context mechanisms. The primacy model (Page & Norris, 1995, figure 7) also demonstrates the ability to reproduce a set of memory

span functions along with the linear memory span against articulation rate function of Hulme, Maughan and Brown (1991, cited in Page & Norris, 1995).

OSCAR was also fitted to the family of reverse 'S' shaped memory span functions of Crannell and Parrish (figure 7.2). The learning rate parameter, corresponding to the strength of encoding, was varied in order to model the effects of stimulus modality on memory span. The limited capacity of the Hebbian matrix and the effect of output interference was reflected in these list length effects.

Phonemic similarity

The phonemic similarity effect is exhibited by a number of models of association and serial order. In particular, when items are represented as distributed vectors of features and acoustic similarity can be modelled by an overlap of vector features. For example, when Metcalfe Eich (1982) presents CHARM with a series of paired associates drawn from a lexicon of similar items, she reports that performance is worse for confusable than nonconfusable stimuli (Conrad, 1964). Lewandowsky and Murdock (1989) observe a phonemic similarity effect with the open-loop implementation of TODAM in a memory span task (Lewandowsky & Murdock, 1989, figure 17). Baddeley, Papagno and Norris (1991, figure 10.3) also demonstrate that an increase in similarity between stimuli reduces performance for the serial position curve. A phonemic similarity effect is also apparent in the network model of the articulatory loop for memory span (Burgess & Hitch, 1992, figure 5) and also serial position curves (Burgess & Hitch, 1992, figures 9 & 10).

Baddeley (1968, experiment 5) describes an experiment in which phonemically confusable items are sandwiched between phonemically nonconfusable items. When subjects are presented with lists of alternately phonemically confusable and nonconfusable items, the serial position curves produced have a "sawtooth" form. Performance is good for nonconfusable items and poor for confusable items. Performance for the nonconfusable items is not affected by the proximity of the confusable items in the alternating list conditions. This is a significant effect as it is not predicted by chaining based accounts of memory. Baddeley, Papagno and Norris

(1991, figure 10.3) attempt to fit TODAM to this effect, but without success. Burgess and Hitch (1992, figure 15) also fail to reproduce this result with the half-chaining half-context driven model of the articulatory loop, although a refined version of the model (Burgess, 1995; Burgess & Hitch, 1996, figure 3.4) does replicate the result. However, Henson, Norris, Page and Baddeley (1996) fit the primacy model (Page & Norris, 1995) to the empirical data, and also provide transposition gradient based analysis of empirical findings which confirm that order errors occur within confusably-like groups.

The acoustic similarity effect for memory span (Baddeley, 1966) was replicated by OSCAR (figure 7.3). OSCAR also provided a reasonable fit to the phonemic similarity effect of Baddeley (1968, experiment 5), although performance for the nonconfusable items suffered when in the presence of confusable items in the alternating list conditions (figure 7.10). Henson, Norris, Page & Baddeley (1996) observe that a similar effect is found in the empirical data as a result of knock-on effects (i.e. poorer performance for nonconfusable items that follow an erroneously recalled confusable item). Clearly representing item confusability by overlapping features provides a suitable means for distributed models to model acoustic similarity. The amount of overlap between each confusable item vector controls the degree to which performance is reduced by similarity. The OSCAR replication of Baddeley's experiment 5 (1968) was also affected by the number of lexicon items available during recall and the context distinctiveness. For example, if fewer distractor items had been available during recall, we anticipate that more order errors would have occurred.

Vocabulary size effects

Many models limit the size of the lexicon of items available for selection during recall. Drewnowski (1980) observes that increasing the size of the lexicon of recallable items does not influence performance. OSCAR was shown to produce a similar effect for different list length conditions (figure 7.4). Clearly this is in contrast to TODAM (Lewandowsky & Murdock, 1989) where the number of recallable items available to the model controls the degree of recency exhibited during serial recall tasks (Mewhort, Popham & James, 1994; Nairne & Neath, 1994).

Distance functions

Burgess and Hitch (1992, figure 11) observe that when the network model of the articulatory loop is primarily chaining based, transpositions involve similar proportions of each remaining list item. However, when the model is primarily context driven, transpositions mainly involve those items immediately adjacent to the target item in accordance with Estes' (1972) observation. The four parameter primacy model (Page & Norris, 1995, figure 7) attempts to fit transposition gradients from Henson, Norris, Page and Baddeley (1996). However, very few transpositions occur between items in the serial positions immediately adjacent to the target position with the result that the distance functions are too sharp.

A natural distribution of order errors about the target serial position was shown to occur in the parameter free version of OSCAR as a consequence of the item similarity and the context distinctiveness (figure 5.16). If the contexts were "highly-distinct" then few order errors occurred, however if they were "less-distinct", and hence more easily confused, more order errors occurred and the transposition gradients become spread out. Item similarity also affected the transposition gradient: confusable items produced more order errors. OSCAR's distance functions adhered to the locality constraint (Estes, 1972) which states that transposition errors involve items from those serial positions immediately adjacent to the target serial position. We demonstrated that where the stimuli were confusable, order errors occurred in large proportions (figures 7.6 & 7.8). Furthermore, in the alternating list conditions of the Baddeley experiment 5 (1968) simulation, order errors were shown to occur within confusably-like groups (figure 7.11).

Item versus order effects

Few recent models of serial order have attempted to address Healy's (1974) observation that order errors produce a bowed serial position curve whilst item errors occur equally across each serial position. Conrad's (1965) suggestion, that order errors result from pairs of item errors, clearly cannot account for this difference. However, using OSCAR we replicated both the *item only* and *order only* error distributions observed by Healy (figure 7.14). This is significant as Healy suggested that different

mechanisms must be responsible for maintaining item and order information. However, OSCAR stores both item and order information at the same location in the composite memory matrix and uses the same recall cue (the appropriate context vector) for both conditions. Clearly, OSCAR provides a single mechanism capable of reproducing the different item and order error distribution illustrated by the data.

Partial reinstatement of context

We also demonstrated the effect of only partially reinstating the learned-context during recall (figure 7.17). The reinstated contexts overlapped with the state of the context at the end of the list. This introduced recency and compared favourably with a probe recall task where ordered recall was not required (Murdock, 1968). No recent model of serial order has considered partial reinstatement of context.

8.6 Developmental model of immediate memory for serial order

As has already been stated, there is clearly a need for a developmental account of memory. In order to address the developmental nature of serial order paradigms such as memory span (e.g. Gathercole & Adams, 1993; Hitch, Halliday & Littler, 1993; Hulme, Thomson, Muir & Lawrence, 1984; Hulme & Tordoff, 1989), the model needs to be able to perform single trial learning with varying degrees of accuracy.

This ability was demonstrated by DARNET when applied to the problem of paired associate learning and recall. DARNET was tested with a partially learned set of storage and retrieval weights, corresponding to an inaccurate single trial learning mechanism. It was later tested with a well learned set of weights which corresponded to a highly accurate single trial learning mechanism. In this manner, and by choosing a wider range of learning abilities, it would be possible to apply the DARNET learning algorithm to the problem of a developmental account of learning.

It was observed that OSCAR requires a learning algorithm capable of performing single trial learning and recall, such as Hebbian association or convolution. Therefore it is feasible that the DARNET learning and recall algorithm could be employed instead of the Hebbian association currently in use.

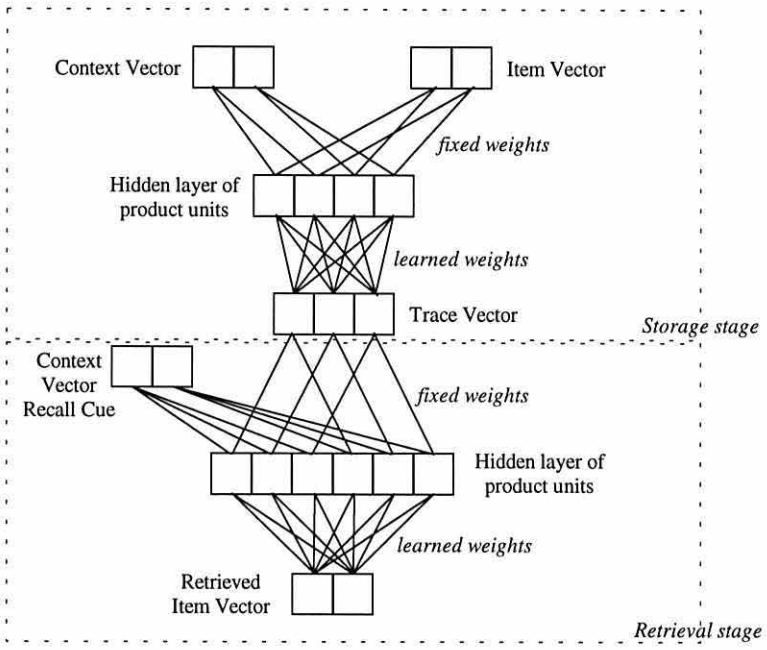

Figure 8.1 *Developmental oscillator based associative recall model*

What is proposed is a refinement of the DARNET learning and recall architecture in which item-to-context associations are stored in the composite memory trace (figure 8.1). An inhibitory process and more complex lexicon of items would have to be included with the refined architecture. Such a network should be capable of producing a developmental account of a range of immediate memory for serial order paradigms.

Future work will see further application of OSCAR to the problem of modelling empirical data and also the integration of OSCAR and DARNET, in order to investigate the developmental nature of short-term memory for serial order.

8.7 Conclusion

To conclude, two novel architectures for immediate memory have been proposed. The first, a developmental associative recall network, DARNET, employed a gradient-descent learning algorithm in order to learn to perform accurate storage and retrieval of novel pairs of vectors. The model demonstrated the ability to perform as accurately as current mathematical models of association, but without the constraints on capacity that they exhibit. More significantly, the use of a connectionist learning algorithm meant that DARNET could provide a developmental account of single trial learning by

examining performance after different amounts of phase-one learning. The model demonstrated how a developmental model may provide an account for current findings.

The second architecture, an oscillator-based associative recall network, OSCAR provided us with a neurobiologically plausible account of serial order learning and recall. The model developed the notion of a time dependent control signal introduced both in other models of immediate memory and elsewhere in other domains. Driven by a set of oscillators, the context was shown to satisfy both the similarity and non-repetition constraints. Items were presented sequentially to the network and were associated to different states of the context signal by the single trial Hebbian learning algorithm. Recall involved reinstating each context in turn and presenting it as a recall cue to the network. The parameter free model demonstrated the ability to produce order errors, such as the bowed serial position curve and the transposition locality gradient, comparable to the empirical data. With the addition of a weights decay, learning rate and noise parameter, the model was shown to fit a range of empirical data including serial order curves, memory span, phonemic similarity, size of vocabulary, transposition gradient effects and item and order error distributions.

APPENDIX

Appendix A: Normalising similar items

One of the arguments levelled at Metcalfe Eich's (1982) simulations using confusable items and CHARM is that it is not clear whether the confusable items remain normalised once the process of introducing the similarity is complete. As normalised stimuli are vital to the retrieval process which implements the dot product as the measure of similarity (Goebel & Lewandowsky, 1991), it is clear that similar confusion should not arise for the DARNET and OSCAR simulations presented here.

The following algorithm implements a method for generating confusable stimuli items which retain their normalisation. In order to build a vocabulary of confusable items, a reference item (the first list item) is produced and normalised as normal. Next, a proportion of the elements of this first vector are copied to the corresponding element positions in each of the remaining vocabulary items. In this manner, each of confusable items contains a proportion of elements identical to those in the reference item, along with values drawn from a normalised distribution about zero with variance of one occupying every remaining vector element. In order to ensure normalisation and confusablity, it is only the features occupying these latter positions that must be updated during the normalisation process.

Before considering the general case, consider an 8-dimensional vector of features, **I**, which contains 4 elements (in the odd positions) which we wish to remain unaffected by the normalisation process:

$$\mathbf{I} = [\, i_1\, i_2\, i_3\, i_4\, i_5\, i_6\, i_7\, i_8 \,] \qquad \text{(A.1)}$$

We know that when the dot product of a normalised vector with itself is calculated, the answer is unity. We wish to find the constant, k, which scales each non-similar element of the vector in order to satisfy this last condition:

$$\mathbf{I} \cdot \mathbf{I} = 1$$

$$\Rightarrow \quad i_1^2 + \left(\frac{i_2}{k}\right)^2 + i_3^2 + \left(\frac{i_4}{k}\right)^2 + i_5^2 + \left(\frac{i_6}{k}\right)^2 + i_7^2 + \left(\frac{i_8}{k}\right)^2 = 1 \qquad \text{(A.2)}$$

Simplifying this equation it is possible to extract k:

$$k = \sqrt{\frac{\left(i_2^2 + i_4^2 + i_6^2 + i_8^2\right)}{1 - \left(i_1^2 + i_3^2 + i_5^2 + i_7^2\right)}} \qquad \text{(A.3)}$$

Therefore, in order to normalise a vector which contains an arbitrary number of elements which must remain unaltered (i.e. because they are the elements in common with the reference vector in a confusable item), each of the unique vector elements must be divided by the constant k. For the current case, where 50% of the reference vector is copied to each of the confusable items, k is calculated by taking the square root of results of the sum of the squares of the "to-be-normalised" elements divided by one minus the sum of the squares of the "elements-in-common". The general case is expressed below:

$$k = \sqrt{\frac{\text{sum of the squares of the "to-be-normalised" elements}}{1 - (\text{sum of the squares of the "elements-in-common"})}} \qquad \text{(A.4)}$$

If confusable vocabularies are generated using this algorithm, we can ensure that items are both similar and normalised.

Appendix B: Methods to ensure reliable data generation

In order to ensure that these simulations are reliable, a number of processes are undertaken to minimise the possibility of erratic results. We have already discussed the manner in which the context vectors are averaged, akin to using a much higher-dimensionality context vector, in order to ensure smoother context similarity profiles. However, a number of other methods are employed in an attempt to eliminate erratic results.

For example, before each sequence of context vectors is presented to the network, a random start position is selected for that context vector set. For example, each set of context vectors contains over a hundred contexts, but for a six item list, with a inter-context spacing of four, only twenty four contexts are required from that set in order to simulate the data. In this case we randomise the start position in order to ensure that for every trial using that set of contexts, we are not just using the first twenty four contexts.

Many of the processes use the computer's ability to generate random numbers (e.g. for selecting values from the normal distribution of features). However, computer random number generators are notoriously *pseudo*-random. Therefore, wherever the computer random number generator is required, it is seeded with a random seed based upon the date and time of simulation. We would anticipate that this would minimise the chance of (e.g.) the same set of item vectors being used in any two simulations.

Furthermore, all of the simulations presented here are the result of a averaging over a number of trials. For a serial recall task, where a serial position curve is required to illustrate the network's performance, typically between 500 and 2000 trials are required in order to ensure that the curve presented is representative of the network's performance. However, as all of the simulations described here have been executed on desktop computers (e.g. Apple Macintosh Power PC or IBM compatible 486DX4-100) often the number of trials executed have been constrained by the requirements of the larger networks.

Appendix C: Further variations on methods for context generation

Simulation 31: *Random delta theta size*

Introduction

Although one of the requirements of the context vector is to employ an array of oscillators with a wide range of frequencies (cf. Church & Broadbent, 1990), in the following experiment, the step for each oscillator phase θ_m after each simulated time step, $\delta\theta_m$ takes a small random value, uniformly distributed between zero and one radians.

Method

An eight element context vector was constructed in the same manner as described previously in section 5.4.2 and illustrated in figure 5.7. Each of θ_m was seeded with a random angle such that the first context vector was at some arbitrary state. The increment in theta at each discrete step, $\delta\theta_m$ was also assigned a small random value uniformly distributed between zero and one. Each of θ_m was incremented by the respective value and the next context vector generated. 32 successive states of the context were calculated and the similarity between averaged recorded over 100 different trials.

Results

The similarity curves presented in figure C.1 clearly illustrate that the context vector formed in this manner reduces in similarity as the temporal separation increases. There is very little noise in the similarity curve once the context are more than 15 steps apart.

Discussion

The contexts formed in this manner clearly satisfy the non-repetition and similarity requirements of a suitable context vector. However, the problem with the context vector in this case is the random distribution of phases across the context elements. Fast and slower evolving context vector elements are not clustered at opposite ends of the vector: instead, randomly distributed across each element. However, this system of

coupled oscillators does appear to produce a very well organised set of contexts (Strogatz & Stewart, 1993) and contexts generated in this form could be used as an idealised set of contexts.

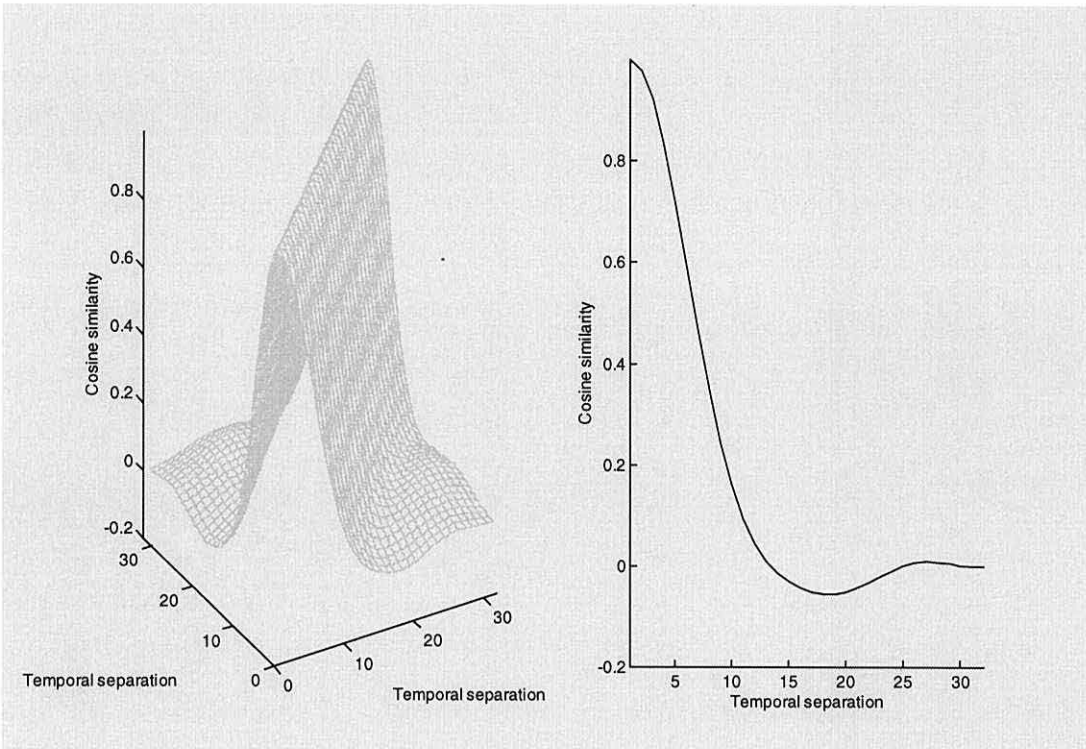

Figure C.1 *End-on and cut-away view of cosine between neighbouring context vectors*

Simulation 32: *Alternate method for context generation*

Introduction

In the following experiment, the context vector elements were constructed in a different manner to that described in 5.4.2. Instead of the distribution of components illustrated in figure 5.7, the following organisation was implemented.

Method

An eight element context vector was constructed in the manner illustrated in figure C.2. Each of θ_m was seeded with a random angle such that the first context vector was at some arbitrary state. The increment in theta at each discrete step, $\delta\theta_m$ was also assigned a small random value uniformly distributed between zero and one. 32

successive states of the context were calculated and the similarity between averaged recorded over 100 different trials.

c(1)	=	$\cos(\theta_1)$	*	$\cos(\theta_2)$	*	$\cos(\theta_3)$
c(2)	=	$\cos(\theta_1)$	*	$\cos(\theta_2)$	*	$\sin(\theta_3)$
c(3)	=	$\cos(\theta_1)$	*	$\sin(\theta_2)$	*	$\cos(\theta_3)$
c(4)	=	$\cos(\theta_1)$	*	$\sin(\theta_2)$	*	$\sin(\theta_3)$
c(5)	=	$\sin(\theta_1)$	*	$\cos(\theta_2)$	*	$\cos(\theta_3)$
c(6)	=	$\sin(\theta_1)$	*	$\cos(\theta_2)$	*	$\sin(\theta_3)$
c(7)	=	$\sin(\theta_1)$	*	$\sin(\theta_2)$	*	$\cos(\theta_3)$
c(8)	=	$\sin(\theta_1)$	*	$\sin(\theta_2)$	*	$\sin(\theta_3)$

Figure C.2 *Composition of the eight element context vector,* ***c***

Results

The similarity function for the contexts generated with this new distribution is presented in figure C.3. This reveals that once again, the similarity curve decreases rapidly towards zero and again oscillates at between small values of approximately zero.

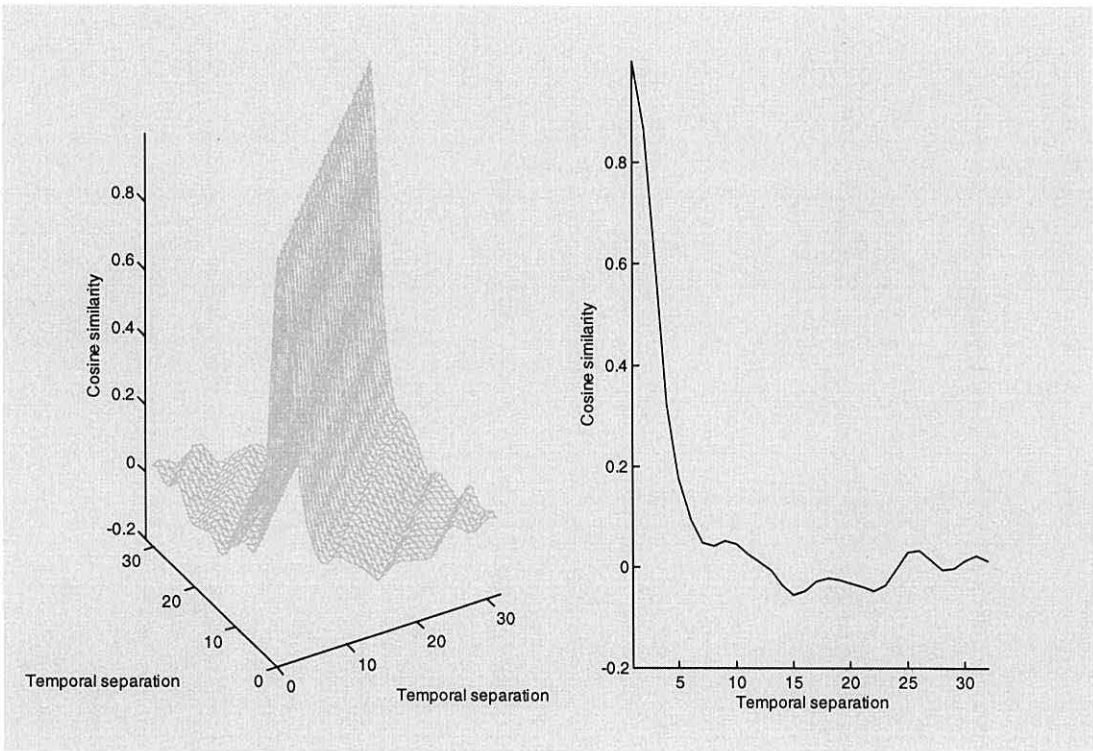

Figure C.3 *End-on and cut-away view of cosine between neighbouring context vectors*

Discussion

Again, this appears to be a suitable method for generating context vectors: these satisfy the non-repetition and similarity properties of the context signal. However, once again the frequencies of the oscillators that form the components of the context vector elements are not widely distributed and it is for this reason that the 2^m relation (section 5.3.3) is chosen as the most suitable distribution for context vector generation, even though the similarity function may not appear to be as desirable as those illustrated elsewhere.

Appendix D: Convolution and correlation as associative mechanisms

Introduction

As was discussed in Chapter 3, a number of successful models of association and serial order have relied upon the mathematical processes of convolution and correlation for item association (e.g. CHARM: Metcalfe Eich, 1982; TODAM: Murdock, 1982). In this appendix, we briefly review these associative mechanisms, although for a full description of these, we refer the reader to Metcalfe Eich (1982; see the Appendix).

Convolution

The convolution operation combines two item vectors into a common memory trace vector. For example, if we consider the convolution of the three-element item vectors **a** and **b** which are represented as distributed vectors of random scalar elements:

$$\mathbf{a} = [\ a_1\ a_2\ a_3\ ]$$
$$\mathbf{b} = [\ b_1\ b_2\ b_3\ ] \qquad \text{(D.1)}$$

The convolution, the trace vector **t**, is a five-element vector that contains information from both of the to-be-learned items. The dimensionality of the convolution trace is fixed and is determined by the equation:

$$n_t = 2n_i - 1 \qquad \text{(D.2)}$$

Where n_t corresponds to the dimensionality of the trace vector and n_i to the dimensionality of the stimuli items. Two properties of the trace vector are that it bears no resemblance to either of the original stimuli items and that an approximation to the second of the pair of item vectors may be retrieved from the association when either of the vectors is presented as a probe.

The trace vector, **t**, is composed as illustrated in equation D.3. Each element consists of the sum of a number of products between elements of both stimuli vectors.

$$t_1 = a_1b_1$$
$$t_2 = a_1b_2 + a_2b_1$$
$$t_3 = a_1b_3 + a_2b_2 + a_3b_1$$
$$t_4 = a_2b_3 + a_3b_2$$
$$t_5 = a_3b_3 \qquad \text{(D.3)}$$

In order that an item may be recovered from this convolution memory trace, one of the items must be presented to the memory trace, and the reverse process of *correlation* performed in order that an approximation of the second item be generated at the output.

Correlation

For the current example, we will assume that item **a** has been selected as the probe item and has been correlated with the memory trace vector, **t**. The resultant output vector, **b'**, is a n_i dimensionality vector:

$$b'_1 = a_3t_1 + a_2t_2 + a_1t_3$$
$$b'_2 = a_3t_2 + a_3t_3 + a_3t_4$$
$$b'_3 = a_3t_3 + a_3t_4 + a_3t_5 \qquad \text{(D.4)}$$

It is clear from this last equation that each element of the output vector contains the sum of a number of products between elements of the probe item and elements of the trace vector. Metcalfe Eich (1982, p632) demonstrates how the output vector, **b'**, can become the to-be-retrieved stimulus item, **b**, and that normalising each stimuli item can improve accuracy during retrieval and recall.

Conclusion

Convolution and correlation provide a formally specified method of single trial association and retrieval. They have formed the basis of a number of successful models of association. However, the capacity of the convolution memory trace is limited by the manner in which the trace dimensionality relates to that of the stimuli items.

REFERENCES

Aschoff, J. (1981). A survey on biological rhythms. In J. Aschoff (Ed.), Handbook of behavioral neurobiology, Vol. 4, Biological Rhythms. New York: Plenum Press.

Baddeley, A. D. (1966). Short-term memory for word sequences as a function of acoustic, semantic and formal similarity. Quarterly Journal of Experimental Psychology, 18, 362-365.

Baddeley, A. D. (1968). How does acoustic similarity influence short-term memory? Quarterly Journal of Experimental Psychology, 20, 249-264.

Baddeley, A. D. (1986). Working Memory. Oxford: Clarendon Press.

Baddeley, A. D., Conrad, R., & Hull, A. J. (1965). Predictability and immediate memory for consonant sequences. Quarterly Journal of Experimental Psychology, 17, 175-177.

Baddeley, A. D., & Hitch, G. J. (1974). Working memory. In G. H. Bower (Ed.), Recent advances in the psychology of learning and motivation (Vol. VIII, pp.47-90). New York: Academic Press.

Baddeley, A. D., Lewis, V. J., & Vallar, G. (1984). Exploring the articulatory loop. Quarterly Journal of Experimental Psychology, 36, 233-252.

Baddeley, A. D., Papagno, C., & Norris, D. (1991). Phonological memory and serial order: A sandwich for TODAM. In W.E. Hockley, & S. Lewandowsky (Eds.), Relating theory and data: Essays on human memory in honor of Bennet B. Murdock, (pp. 175-194). Hillsdale, NJ: Erlbaum.

Baddeley, A. D., Thomson, N., & Buchanan, M. (1975). Word length and the structure of short-term memory. Journal of Verbal Learning and Verbal Behaviour, 14, 575-589.

Beling, I. (1929). Über das Zeitgedächtnis der Bienen. Zeitschrift für vergleichende Physiologie, 9, 259-338.

Bjork, E. L. & Healy, A. F. (1974). Short-term order and item retention. Journal of Verbal Learning and Verbal Behavior, 13, 80-97.

Brown, G. D. A., Dalloz, P., & Hulme, C. (1995). Mathematical and connectionist models of human memory: A comparison. Memory, 3(2), 113-145.

Brown, G. D. A., Hulme, C., & Dalloz, P. (1996). Modelling human memory: Connectionism and convolution. British Journal of Mathematical and Statistical Psychology, 49, 1-24.

Brown, G. D. A., Hyland, P., & Hulme, C. (1994). The effects of varying memory vector size in a network that learns to learn. Proceedings of the World Congress on Computational Intelligence, IV, 2291-96. Piscataway, New Jersey: IEEE.

Brown, G. D. A., Preece, T., & Hulme, C. (1995). Learning to learn in a connectionist network: the development of associative learning. In J. P. Levy, D. Bairaktaris, J. A. Bullinaria, & P. Cairns (Eds.), Connectionist models of memory and language, (pp. 41-56). UCL Press: London.

Burgess, N. (1995). A solvable connectionist model of immediate recall of ordered lists. In G. Tesauro, D. Touretzky, & J. Alspector (Eds.), Neural Information Processing Systems 7, San Mateo, CA: Morgan Kaufmann.

Burgess, N., & Hitch, G. J. (1992). Toward a network model of the articulatory loop. Journal of Memory and Language, 31(4), 429-460.

Burgess, N., & Hitch, G. J. (1996). A connectionist model of STM for serial order. In S. E. Gathercole (Ed.), Models of short-term memory (pp. 51-72). Hove: Psychology Press.

Chater, N. & Conkey, P. (1994). Sequence processing with recurrent neural networks. In M. Oaksford, & G. D. A. Brown (Eds.), Neurodynamics and psychology. London: Academic Press.

Church, R. M., & Broadbent, H. A. (1990). Alternative representations of time, number, and rate. Cognition, 37, 55-81.

Conrad, R. (1959). Errors of immediate memory. British Journal of Psychology, 50, 349-359.

Conrad, R. (1960a). Serial order intrusions in immediate memory. British Journal of Psychology, 51(1), 45-48.

Conrad, R. (1960b). Very brief delay of immediate recall. Quarterly Journal of Experimental Psychology, 12, 45-47.

Conrad, R. (1964). Acoustic confusions in immediate memory. British Journal of Psychology, 55(1), 75-84.

Conrad, R. (1965). Order error in immediate recall of sequences. Journal of Verbal Learning and Verbal Behavior, 4, 161-169.

Conrad, R., & Hull, A. J. (1964). Information, acoustic confusion and memory span. British Journal of Psychology, 55(4), 429-432.

Conrad, R., & Hull, A. J. (1968). Input modality and the serial position curve in short-term memory. Psychonomic Science, 10(4), 135-136.

Crannell, C. W., & Parrish, J. M. (1957) A comparison of immediate memory span for digits, letters and words. Journal of Psychology, 44, 319-327.

Dehaene, S. (1993). Temporal oscillations in human perception. Psychological Science, 4(4), 264-270.

Drewnowski, A. (1980). Attributes and Priorities in Short-Term Recall: A New Model of Memory Span. Journal of Experimental Psychology: General , 109(2), 208-250.

Ebbinghaus, H. (1913). Memory: A contribution to experimental psychology. New York: Teachers College, Columbia University. (Reprinted by Dover, 1964).

Elman, J. L. (1990). Finding structure in time. Cognitive Science, 14, 179-211.

Estes, W. K. (1972). An associative basis for coding and organization in memory. In A.W. Melton, & E. Martin (Eds.), Coding processes in human memory, (pp. 161-190). Washington, DC: Winston.

Ellis, N. C., & Hennelley, R. A. (1980). A bilingual word-length effect: Implications for intelligence testing and the relative ease of mental calculation in Welsh and English. British Journal of Psychology, 71, 43-52.

Fuchs, A. H. (1969). Recall for order and content of serial word lists in short-term memory. Journal of Experimental Psychology, 82(1), 14-21.

Gallistel, C. R. (1990). The organization of learning. Cambridge, MA: MIT Press/Bradford Books.

Gathercole, S. E., & Adams, M. A. (1993). Phonological working memory in very young children. Developmental Psychology, 29(4), 770-778.

Gill, F. B. (1988). Traplining foraging by hermit hummingbirds: Competition for an undefended renewable source. Ecology, 69, 1933-1942

Goebel, R. P. & Lewandowsky, S. (1991). Retrieval measures in distributed memory models. In W. E. Hockley, & S. Lewandowsky (Eds.), Relating theory and data: Essays on human memory in honor of Bennet B. Murdock, (pp. 509-527). Hillsdale, NJ: Erlbaum.

Haykin, S. (1994). Neural Networks: A Comprehensive Foundation. New York: Macmillan College Publishing Company.

Healy, A. F. (1971). Short-term memory of consonant order. In Communications in Mathematical Psychology, Rockerfeller University Technical Reports.

Healy, A. F. (1974). Separating item from order information in short-term memory. Journal of Verbal Learning and Verbal Behavior, 13, 644-655.

Hebb, D. O. (1961). Distinctive features of learning in the higher animal. In J. E. Delafresnaye (Ed.), Brain Mechanisms and Learning, (pp. 37-46). New York: Oxford University Press.

Henson, R. N. A. (in press). Item repetition in short-term serial recall: Ranschburg repeated.

Henson, R. N. A., Norris, D. G., Page, M. P. A., & Baddeley, A. D. (1996). Unchained memory: error patterns rule out chaining models of immediate, serial recall. Quarterly Journal of Experimental Psychology, 49A, 80-115.

Hitch, G. J., Halliday, M. S., & Littler, J. E. (1993). Development of memory span for spoken words: The role of rehearsal and item identification processes. British Journal of Developmental Psychology, 11, 159-169.

Houghton, G. (1990). The problem of serial order: A neural network model of sequence learning and recall. In R. Dale, C. Mellish, & M. Zock (Eds.), Current research in natural language generation, London: Academic Press.

Houghton, G. (1994a). Inhibitory control of neurodynamics: Opponent mechanisms in sequencing and selective attention. In M. Oaksford & G. D. A. Brown (Eds.), Neurodynamics and psychology. London: Academic Press.

Houghton, G. (1994b). Some formal variations on the theme of competitive queueing. (Tech. Rep. UCL-PSY-CQ1). University College London.

Houghton, G., Glasspool, D.W. & Shallice, T. (1994). Spelling and serial recall: Insights from a competitive queueing model. In G. D. A. Brown, & N. C. Ellis (Eds.), Handbook of spelling: Theory, process and intervention: John Wiley & Sons Ltd.

Houghton, G., & Hartley, T. (1995). Parallel models of serial behaviour: Lashley revisited. Psyche, 2, 25.

Hulme, C., Thomson, N., Muir, C., & Lawrence, A. (1984). Speech rate and the development of short-term memory span. Journal of Experimental Child Psychology, 38, 241-253.

Hulme, C. & Tordoff, V. (1989). Working memory development: The effects of speech rate, word length and acoustic similarity on serial recall. Journal of Experimental Child Psychology, 47, 72-87.

Hulme, C., Maughan, S., & Brown, G. D. A. (1991). Memory for familiar and unfamiliar words: Evidence for a long term contribution to short-term memory span. Journal of Memory and Language, 30, 685-701.

Jahnke, J. C. (1963). Serial position effects in immediate serial recall. Journal of Verbal Learning and Verbal Behavior, 2, 284-287.

Jahnke, J. C. (1969). The Ranschburg effect. Psychological Review, 76, 592-605.

Johnson, G. J. (1991). A distinctiveness model of serial learning. Psychological Review, 98(2), 204-217.

Johnson, N. F. (1969). Chunking: Associative chaining versus coding. Journal of Verbal Learning and Verbal Behavior, 8, 725-731.

Johnson, N. F. (1970). The role of chunking and organization in the process of recall. In G. H. Bower (Ed.), The Psychology of Learning and Motivation Vol. 4, (pp. 171-247). New York: Academic Press.

Jordan, M. I. (1986). Serial order: A parallel, distributed approach. (Tech. Rep. No. 8604). La Jolla: University of California, San Diego.

Lashley, K. S. (1951). The problem of serial order in behavior. In L. A. Jeffress (Ed.), Cerebral mechanisms in behavior: The Hixon Symposium, (pp. 112-136). New York & London: Hafner Publishing Comp.

Latour, P. L. (1967). Evidence of internal clocks in the human operator. Acta Psychologica, 27, 341-348.

Lee, C. L. (1992). The perturbation model of short-term memory: A review and some further developments. In A. F. Healy, S. M. Kosslyn, & R. M. Shiffrin (Eds.), From Learning Processes to Cognitive Processes: Essays in Honour of W.K.Estes Vol. 2, (pp. 119-141). Hillsdale, NJ: Erlbaum.

Lewandowsky, S. (1991). Gradual unlearning and catastrophic interference: A comparison of distributed architectures. In W. E. Hockley, & S. Lewandowsky (Eds.), Relating theory and data: Essays on human memory in honor of Bennet B. Murdock, (pp. 445-476). Hillsdale, N.J.: Erlbaum.

Lewandowsky, S., & Murdock, B. B. (1989). Memory for serial order. Psychological Review, 96(1), 25-57.

Lewandowsky, S., & Shu-Chen Li. (1994). Memory for serial order revisited. Psychological Review, 101(3), 539-543.

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: the sequential learning problem. In G. H. Bower (Ed.), Psychology of Learning and Motivation Vol. 24, (pp. 109-165). New York: Academic Press.

Metcalfe, J. (1991). Recognition failure and the composite memory trace in CHARM. Psychological Review, 98(4), 529-553.

Metcalfe, J., & Murdock, B. B. (1981). An encoding and retrieval model of single-trial free recall. Journal of Verbal Learning and Verbal Behavior, 20, 161-189.

Metcalfe Eich, J. (1982). A composite holographic associative recall model. Psychological Review, 89(6), 627-661.

Metcalfe Eich, J. (1985). Levels of processing, encoding specificity, elaboration, and CHARM. Psychological Review, 92(1), 1-38.

Mewhort, D. J. K., Popham, D., & James, G. (1994). On serial recall: A critique of chaining in the theory of distributed associative memory. Psychological Review, 101(3), 534-538.

Millar, A. J., Carré, I. A., Strayer, C. A., Chua N, & Kay, S. A. (1995). Circadian Clock Mutants in Arabidopsis Identified by Luciferase Imaging. Science, 267, 1161-1163.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. The Psychological Review, 63(2), 81-97.

Murdock, B. B. (1968). Serial order effects in short-term memory. Journal of Experimental Psychology Monograph Supplement, 76(Pt. 2), 1-15.

Murdock, B. B. (1974). Human memory: Theory and data. Potomac, Maryland: Lawrence Erlbaum Associates.

Murdock, B. B. (1982). A theory for the storage and retrieval of item and associative information. Psychological Review, 89(6), 609-625.

Murdock, B. B. (1983). A distributed memory model for serial-order information. Psychological Review, 90(4), 316-338.

Murdock, B. B. (1992). Serial organization in a distributed memory model. In A. F. Healy, S. M. Kosslyn, & R. M. Shiffrin (Eds.), From Learning Theory to Connectionist Theory: Essays in Honor of William K. Estes Vol. 1, (pp. 201-225). Hillsdale, NJ: Erlbaum.

Murdock, B. B. (1993). TODAM2: A model for the storage and retrieval of item, associative, and serial-order information. Psychological Review, 100(2), 183-203.

Murray, D. J. (1965). Vocalization-at-presentation and immediate recall, with varying presentation-rates. Quarterly Journal of Experimental Psychology, 17, 47-56.

Murray, D. J. (1967). The role of speech responses in short-term memory. Canadian Journal of Psychology, 21(3), 263-276.

Murray, D. J. (1968). Articulation and acoustic confusability in short-term memory. Journal of Experimental Psychology, 78(4), 679-684.

Nairne, J. S., & Neath, I. (1994). Critique of the retrieval/deblurring assumptions of the theory of distributed associative memory. Psychological Review, 101(3), 528-533.

Page, M. P. A., & Norris, D. (1995). The Primacy Model: A new model of immediate serial recall. Manuscript submitted for publication.

Ratcliff, R. (1990). Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. Psychological Review, 97(2), 285-307.

Rumelhart, D. E., Hinton G. E. & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds), Parallel Distributed Processing, Volume 1: Foundations, pp318-362: Cambridge, Mass.: MIT Press.

Rumelhart, D. E., & McClelland, J. L. (1986). On Learning the Past Tenses of English Verbs. In J. L. McClelland & D. E. Rumelhart (Eds), Parallel Distributed Processing, Volume 2: Psychological and biological models, pp216-271: Cambridge, Mass.: MIT Press.

Shiffrin, R. M., & Cook, J. R. (1978). Short-term forgetting of item and order information. Journal of Verbal Learning and Verbal Behavior, 17, 189-218.

Strogatz, S. H., & Stewart, I. (1993). Coupled oscillators and biological synchronization. Scientific American, 269(6), 68-75.

Treisman, M., Faulkner, A., Naish, P. L. N., & Brogan, D. (1990). The internal clock: evidence for a temporal oscillator underlying time perception with some estimates of its characteristic frequency. Perception, 19, 705-743.

Wickelgren, W. A. (1964). Size of rehearsal group and short-term memory. Journal of Experimental Psychology, 68(4), 413-419.

Wickelgren, W. A. (1965a). Short-term memory for phonemically similar lists. American Journal of Psychology, 78, 567-74.

Wickelgren, W. A. (1965b). Acoustic similarity and intrusion errors in short-term memory. Journal of Experimental Psychology, 70(1), 102-108.

Wickelgren, W. A. (1965c). Short-term memory for repeated and non-repeated items. Quarterly Journal of Experimental Psychology, 17, 14-25.

Wickelgren, W. A. (1966). Associative intrusions in short-term recall. Journal of Experimental Psychology, 72(6), 853-858.

Wickelgren, W. A. (1967). Rehearsal grouping and hierarchical organization of serial position cues in short-term memory. Quarterly Journal of Experimental Psychology, *19*, 97-102.

Wickelgren, W. A. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. Psychological Review, *76*, 1-15.